COMPUTAÇÃO & O MUNDO

9°Ano

Livro do Professor

Luis Gustavo de Jesus Araujo
Bianca Leite Santana
Roberto Almeida Bittencourt

Livro do Professor - 9° ano

Projeto Computação Fundamental

Luis Gustavo de Jesus Araujo
Bianca Leite Santana
Roberto Almeida Bittencourt

https://sites.google.com/view/computacaofundamental/

Versão 1.0

Versão 1.0

Autores: Luis Gustavo de Jesus Araujo, Bianca Leite Santana e Roberto Almeida Bittencourt

Neste livro, nas aulas da Unidade I, utilizamos imagens de comandos e personagens do Scratch, que é um projeto da Fundação Scratch em colaboração com o grupo Lifelong Kindergarten no MIT Media Lab e está disponível gratuitamente em https://scratch.mit.edu. Tais imagens estão disponíveis sob a licença Attribution-ShareAlike 2.0 International (CC BY-SA 2.0). Nas aulas da Unidade IV, utilizamos imagens de comandos do mBlock, que é um projeto inteiramente baseado em Scratch e mantido pela Makeblock e está disponível gratuitamente em https://www.mblock.cc/en-us/. Também nas aulas da Unidade IV, usamos a plataforma de hardware livre Arduino, cujos dispositivos estão licenciados através da licença Attribution-ShareAlike (CC BY-SA). Algumas das aulas deste livro baseiam-se em ideias disponíveis em atividades do livro *Unplugged Computing... off-line activities and games for all ages* (1998), por Tim Bell, Ian H. Witten e Mike Fellows.

FICHA CATALOGRÁFICA

A663c Araujo, Luis Gustavo de Jesus.
Computação e o mundo: livro do professor / Luis Gustavo de Jesus Araujo, Bianca Leite Santana, Roberto Almeida Bittencourt. – Feira de Santana: [s.n.], 2020.
169p.: il.

Edição do autor.
ISBN: 978-65-00-01451-8

1. Ciência da Computação 2. Programação de Computadores. 3. Pensamento Computacional I. Título

CDD: 004
CDU: 004

Sumário

Sobre o Livro 01
Visão Geral 06

UNIDADE I – Simulações

Aula 01 – Impacto da Tecnologia na Cultura Humana 10
Aula 02 – Minha nova fábrica de chocolate 13
Aula 03 – A matemática do movimento 20
Aula 04 – Modelo Atômico de Bohr 25
Aula 05 – Sistema Solar 32
Aula 06 – Fazendo seu Projeto de Ciências (parte 1) 37
Aula 07 – Fazendo seu Projeto de Ciências (parte 2) 42

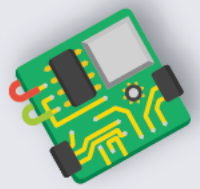
UNIDADE II – Manipulação de Imagens

AULA 1 – Pixel a Pixel 47
AULA 2 – Fazendo Seus Próprios Filtros 49
AULA 3 - Mais Efeitos Utilizando Matemática 53
AULA 4 - Mais efeitos visuais 58
AULA 5 - Brincando com Chroma key 62
AULA 6 - Gerador de Memes 65
AULA 7 – Cidade na Lama 68

UNIDADE III – Python

AULA 1 – Fluxograma 73
AULA 2 – Apertadinho 81
AULA 3 – Enquetes 85
AULA 4 – Criptografia de César 89
AULA 5 – Jogo da Forca 97
AULA 6 – A Droga da Obediência 103
AULA 7 – Ovos da Páscoa 107
AULA 8 – Jogo da Velha 111

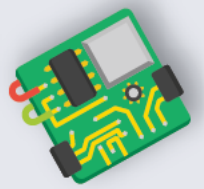
UNIDADE IV – Arduino

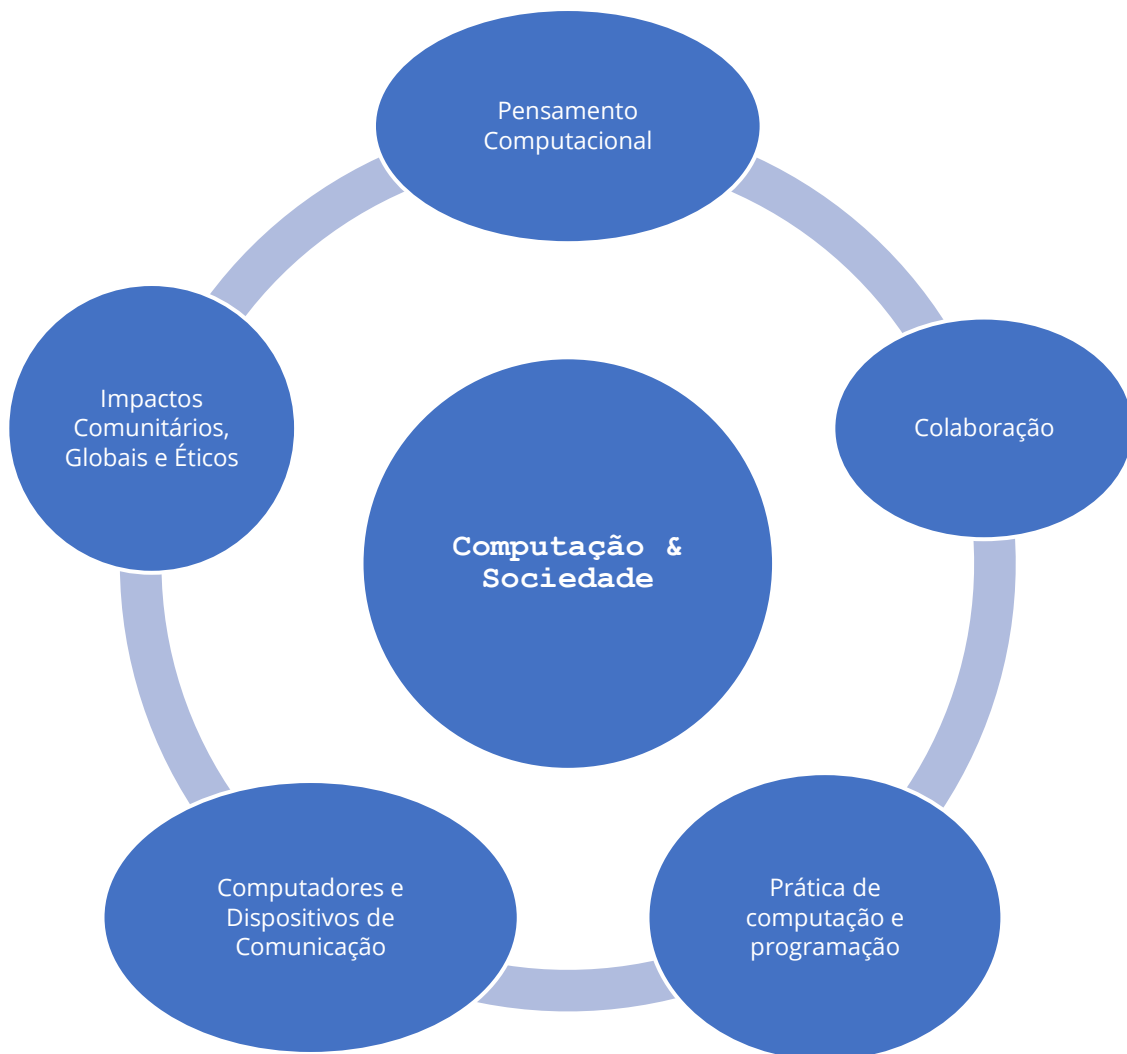
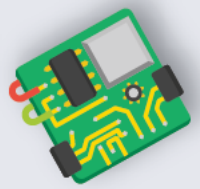
AULA 1 - Construindo um Semáforo 118
AULA 2 – Medindo a Luminosidade em um Ambiente 126
AULA 3 – Criando um Piano com o Arduino 132
AULA 4 – Criando um Robô Dançarino 139
AULA 5 – Criando um Robô Guardião do Tesouro 145
AULA 6 – Como funciona o Computador? 151
AULA 7 – Criando seu Primeiro Protótipo com o Arduino 159
AULA 8 – Apresentando seu Primeiro Protótipo com o Arduino 163

Sobre o Livro

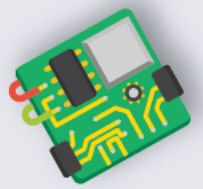
Este livro é concebido para os estudantes do nono ano do ensino fundamental. O objetivo principal deste livro é conectar os conhecimentos dos estudantes sobre a computação com o mundo em que vivem, além de aprofundar o uso da computação para expressar ideias na forma de software ou de hardware através da solução criativa de problemas de interesse dos estudantes e do mundo.

Na primeira unidade, iniciamos a conexão entre os conhecimentos prévios de pensamento computacional e programação em linguagens de blocos com o mundo das ciências naturais. Atividades de simulação de fenômenos científicos são propostas para os estudantes, permitindo aplicar seus conhecimentos prévios na expressão matemática e lógica de problemas do mundo real. No final da unidade, os estudantes fazem um projeto de simulação de ciências de seus próprios interesses.

Na segunda e terceira unidades, os estudantes fortalecem seu domínio do uso de uma linguagem de programação textual (Python), já iniciado no ano anterior através do contexto metodológico de figuras geométricas e da biblioteca Turtle. Na segunda unidade, o contexto metodológico de imagens digitais permite que eles reapliquem os conceitos de sequência, seleção, repetição e funções, adicionando algumas estruturas de dados básicas como vetores unidimensionais e bidimensionais. As imagens digitais permitem o feedback imediato sobre a execução do código do mesmo modo que o palco do Scratch e a tela do Turtle. Na terceira unidade, movemos para o contexto metodológico mais tradicional de programação usando entradas e saídas textuais. A linguagem permanece Python, para manter a familiaridade, e os estudantes aprendem a usar variáveis inteiras, reais e textuais, expressões lógicas e aritméticas e as estruturas de programação e de dados previamente aprendidas para construir jogos e pequenos programas. O uso de strings é potencializado com jogos de palavras. Assim, é possível se aproximar de um contexto mais tradicional, mas mantendo uma abordagem lúdica.

Finalmente, na quarta unidade, introduzimos o contexto metodológico de hardware livre através da plataforma Arduino e de projetos básicos de eletrônica digital. Com este contexto, os estudantes podem fazer seus

programas em linguagem de blocos comandar dispositivos eletrônicos e fazer simulações de situações realísticas como as existentes em cidades, envolvendo sensores, processadores e atuadores. A conexão com a robótica educacional trabalhada no ano anterior é natural, já que vários dos dispositivos usados se assemelham aos integrados à plataforma de robô usada no ano anterior. Do mesmo modo, os custos escolares são mantidos sob controle com os dispositivos Arduino e componentes eletrônicos de baixo custo.

Ao final de um ano com a proposta Computação e o Mundo, os estudantes exercitam ainda mais os eixos de pensamento computacional, colaboração, práticas de computação e programação e de computadores e dispositivos de comunicação, integrando-os ao mundo em que vivem, além de continuar a discutir os impactos comunitários, globais e éticos da computação.

Pensamento Computacional (PC)

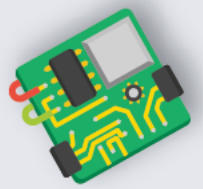
PC1. Descrever como uma simulação pode ser usada para resolver um problema.
PC2. Usar passos básicos na solução de problemas algorítmicos para projetar soluções (por exemplo, declaração e exploração de problemas, exemplos de instâncias, design, implementação de uma solução, teste, avaliação).
PC4. Definir um algoritmo como uma sequência de instruções que pode ser processada por um computador.
PC5. Avaliar maneiras pelas quais diferentes algoritmos podem ser usados para resolver o mesmo problema.
PC6. Trabalhar com algoritmos de busca e classificação.
PC7. Descrever e analisar uma sequência de instruções que estão sendo seguidas (por exemplo, descrever o comportamento de um personagem em um videogame conforme orientado por regras e algoritmos).
PC8. Representar dados de várias formas, incluindo texto, sons, imagens e números.
PC9. Usar representações visuais de estados de problemas, estruturas e dados (por exemplo, gráficos, diagramas, diagramas de rede, fluxogramas).
PC10. Interagir com modelos e simulações de conteúdo específico (por exemplo, ecossistemas, epidemias, dinâmica molecular) para apoiar a aprendizagem e pesquisa.
PC11. Avaliar quais tipos de problemas podem ser resolvidos usando modelagem e simulação.
PC12. Analisar o grau em que um modelo de computador representa com precisão o mundo real.
PC13. Usar abstração para decompor um problema em subproblemas.
PC14. Compreender a noção de hierarquia e abstração na computação, incluindo linguagens de alto nível, tradução, conjunto de instruções, e circuitos lógicos.
PC15. Examinar as conexões entre os elementos da matemática e da ciência da computação, incluindo números binários, lógica, conjuntos e funções.
PC16. Fornecer exemplos de aplicações interdisciplinares do pensamento computacional.

Práticas de Computação e Programação (PCP)

PCP5. Reunir e manipular dados usando uma variedade de ferramentas digitais.
PCP10. Demonstrar compreensão dos algoritmos e sua aplicação prática.
PCP11. Implementar soluções de problemas usando uma linguagem de programação, incluindo: comportamento de looping, instruções condicionais, lógica, expressões, variáveis e funções.
PCP13. Identificar carreiras interdisciplinares que são aprimoradas pela ciência da computação.
PCP15. Coletar e analisar dados que são produzidos a partir de várias execuções de um programa de computador.

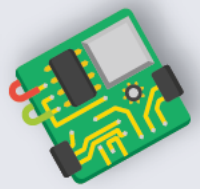
Colaboração (C)

C6. Criar, desenvolver, publicar e apresentar, de forma colaborativa, produtos (por exemplo, vídeos, podcasts, websites) usando recursos de tecnologia que demonstram e comunicam conceitos de currículo.
C7. Colaborar com colegas, especialistas e outras pessoas usando práticas colaborativas, como programação em pares, trabalho em equipes de projeto e participação em atividades de aprendizado ativo em grupo.
C8. Apresentar disposição necessária para colaboração: fornecer feedback útil, integrar feedback, compreender e aceitar múltiplas perspectivas, socialização.

Computadores e Dispositivos de Comunicação (CDC)

CDC1. Reconhecer que os computadores modelam o comportamento inteligente (como encontrado em robótica, reconhecimento de fala e linguagem e animação por computador).
CDC3. Reconhecer que os computadores são dispositivos que executam programas.
CDC4. Identificar uma variedade de dispositivos eletrônicos que contêm processadores computacionais.
CDC5. Demonstrar uma compreensão da relação entre hardware e software.

CDC9. Descrever o que distingue os seres humanos das máquinas, concentrando-se na inteligência humana versus a inteligência das máquinas e nas maneiras pelas quais podemos comunicar.
CDC10. Descrever maneiras pelas quais os computadores usam modelos de comportamento inteligente (por exemplo, movimento do robô, compreensão da fala e da linguagem, e visão computacional).

Impactos Comunitários, Globais e Éticos (IC)

IC3. Discutir questões básicas relacionadas ao uso responsável de tecnologia e informação, e as consequências do uso inadequado.
IC6. Identificar o impacto da tecnologia (por exemplo, redes sociais, cyberbullying, computação e comunicação móvel, tecnologias da Web, segurança cibernética e virtualização) na vida pessoal e na sociedade.
IC10. Demonstrar conhecimento das mudanças nas tecnologias da informação ao longo do tempo e os efeitos que essas mudanças têm sobre a educação, local de trabalho e sociedade.
IC14. Discutir como a distribuição desigual de recursos de computação em uma economia global levanta questões de equidade, acesso e poder.

Visão Geral

As aulas estão divididas em quatro unidades. Embora cada unidade tenha um tema central, tópicos dos cinco eixos de objetivos esperados de aprendizagem são trabalhados de maneira concomitante.

Em cada uma das aulas, os estudantes são incentivados a escreverem um diário de bordo descrevendo suas experiências.

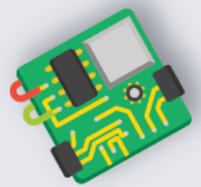
UNIDADE I – Simulações

A primeira unidade do programa, composta por 7 aulas, tem como tema central o desenvolvimento de simulações utilizando o Scratch.

	AULA	CONTEÚDO
UNIDADE I	1 Impacto da Tecnologia na Cultura Humana C6, C7, C8, IC3, IC6, IC10 e IC14	Conceito e Exemplos de Tecnologia; Impacto da Tecnologia na vida humana; Questões éticas e o uso da Tecnologia.
	2 Minha nova fábrica de chocolate PC2, C7 e C8	Design de interfaces; Affordances; Mapeamento; Efeitos de transferência; Estereótipos populacionais; Ícones.
	3 A matemática do movimento PC1, PC2, PC4, PC5, PC7, PC10, PC11, PC12, PC15 e PCP11	Movimento uniformemente variado; Scratch; Variáveis, Entrada e saída, Loops, Condicionais, Contadores, Expressões aritméticas.
	4 Modelo Atômico de Bohr Mesmos objetivos da aula anterior	Modelo atômico de Bohr; Scratch; Variáveis, Entrada e saída, Loops, Condicionais, Contadores, Expressões aritméticas.
	5 Sistema Solar Mesmos objetivos da aula anterior	Sistema Solar; Movimento elíptico; Scratch; Variáveis, Entrada e saída, Loops, Condicionais, Contadores, Expressões aritméticas.
	6 Fazendo seu Projeto de Ciências (parte 1) PC1, PC2, PC4, PC5, PC7, PC10, PC11, PC12, PC13, PC15 e PCP11	Conceitos específicos de ciências; Scratch; Variáveis, Entrada e saída, Loops, Condicionais, Contadores, Expressões aritméticas, Funções matemáticas.
	7 Fazendo seu Projeto de Ciências (parte 2) Mesmos objetivos da aula anterior	

Os estudantes entendem como a matemática pode ser utilizada para criar modelos computacionais que representam fenômenos da natureza.

UNIDADE II – Manipulação de Imagens

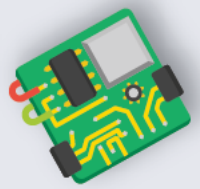
A segunda unidade do programa, composta por 7 aulas, trabalha com a manipulação de imagens através da implementação de programas em linguagem Python.

	AULA	CONTEÚDO
UNIDADE II	1 Pixel a Pixel PC2, PC5, PC8, PC12, PC15 , PCP10 e PCP11	Imagens Digitais; Pixels; Padrão de cores RGB; Python; Variáveis, sequência, loops simples e funções de bibliotecas.
	2 Fazendo Seus Próprios Filtros PC2, PC5, PC8, PC15 , PCP10 e PCP11	Efeitos de imagem; Pixels; Funções do JES; Variáveis, expressões aritméticas, sequência, loops, listas e funções.
	3 Mais Efeitos Utilizando Matemática Mesmos objetivos da aula anterior	
	4 Mais efeitos visuais Mesmos objetivos da aula anterior	
	5 Brincando com Chroma key Mesmos objetivos da aula anterior	Chroma key; Pixels; Funções do JES; Variáveis, expressões aritméticas, condicionais, loops, listas e funções.
	6 Gerador de Memes Mesmos objetivos da aula anterior	Efeitos de imagem; Pixels; Funções do JES; Variáveis, expressões aritméticas, sequência, loops, listas e funções.
	7 Cidade na Lama PC1, PC5, PC6 e PC9	Grafos não dirigidos; Algoritmos sobre grafos.

Os estudantes utilizam o ambiente de desenvolvimento JES para manipular propriedades dos pixels das imagens, implementando efeitos como escala de cinza, negativo, *chroma key*, dentre outros.

UNIDADE III – Programação em Python

A terceira unidade do programa, composta por 8 aulas, busca aprofundar as habilidades de programação dos estudantes com a linguagem Python.

	AULA	CONTEÚDO
UNIDADE III	1 Fluxograma PC2, PC4, PC8, PC9, PC16, PCP5 e PCP10	Fluxogramas: Terminal, Seta de Fluxo, Processamento; Entrada de Dados, Saída de Dados; Decisão; Conector, Loops.
	2 Apertadinho PC2, PC4, PC8, PC9, PC15, PC16, PCP5, PCP10 e PCP11	Python; Variáveis, atribuição, antrada e saída, loops, condicionais, expressões lógicas e relacionais.
	3 Enquetes PC2, PC4, PC8, PC15, PC16, PCP5, PCP10, PCP11 e PCP15	Python; Variáveis, entrada e saída, loops, condicionais, contadores, expressões aritméticas.
	4 Criptografia de César PC2, PC4, PC8, PC15, PC16, PCP5, PCP10 e PCP11	Criptografia; Python; Variáveis, entrada e saída, loops, condicionais, strings, indexação de strings.
	5 Jogo da Forca PC2, PC4, PC8, PC15, PC16, PCP5, PCP10, PCP11 e PCP15	Python; Variáveis, entrada e saída, loops, condicionais, strings, indexação de strings, expressões lógicas.
	6 A Droga da Obediência PC2, PC4, PC8, PC15, PC16, PCP5, PCP10 e PCP11	Criptografia; Python; Variáveis, entrada e saída, loops, condicionais, strings, manipulação de strings.
	7 Ovos da Páscoa PC2, PC4, PC8, PC15, PC16, PCP5, PCP10, PCP11 e PCP15	Python; Variáveis, entrada e saída, loops, condicionais, expressões lógicas e aritméticas, strings, indexação de strings, Arquivos.
	8 Jogo da Velha PC2, PC4, PC8, PC15, PC16, PCP5, PCP10 e PCP11	Python; Variáveis, entrada e saída, loops, condicionais, strings, listas e sua indexação, funções.

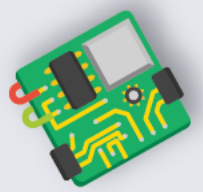
Cada aula implementa projetos de propósito geral, como sistema de enquetes, jogos da força, jogo da velha, dentre outros.

UNIDADE IV - Arduino

A quarta unidade do programa, composta por 8 aulas, apresenta aos estudantes o universo da eletrônica e robótica com Arduino.

UNIDADE IV	AULA	CONTEÚDO
	1 Construindo um Semáforo PC2, PC7, PCP10, PCP11, PCP13, C7, C8, CDC1, CDC3, CDC4, CDC9 e CDC10	Circuitos elétricos e eletrônicos; Arduino; mBlock; Sequências e loops.
	2 Medindo a Luminosidade em um Ambiente Mesmos objetivos da aula anterior	Arduino; Sensor de luminosidade dade LDR; Sinais digitais e analógicos; mBlock; Sequências, loops e condicionais.
	3 Criando um Piano com o Arduino Mesmos objetivos da aula anterior	Arduino; Push button; Buzzer; mBlock; Sequências, loops e condicionais.
	4 Criando um Robô Dançarino Mesmos objetivos da aula anterior	Arduino; Servomotor (Micro Servo 9g SG90 TowerPro); mBlock; Sequências, loops e funções.
	5 Criando um Robô Guardião do Tesouro Mesmos objetivos da aula anterior	Arduino; Sensor ultrassônico (HSR04); mBlock; Sequências, loops e condições.
	6 Como funciona o Computador? PC14, CDC3 e CDC5	Arquitetura de hardware; Arquitetura de software.
	7 Criando seu Primeiro Protótipo com o Arduino Mesmos objetivos da aula 1	Circuitos eletrônicos; Arduino; Componentes; mBlock; Sequências, loops, condições e funções.
	8 Apresentando seu Primeiro Protótipo com o Arduino Mesmos objetivos da aula 1	Circuitos eletrônicos; Arduino; Componentes; mBlock; Sequências, loops, condições e funções; Apresentação de Slides.

Cada aula implementa um projeto diferente, utilizando LEDs, motores, sensores, dentre outros componentes.

UNIDADE I

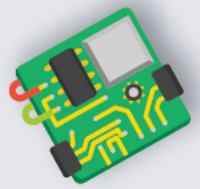
Simulações

AULA 1 - Impacto da Tecnologia na Cultura Humana

Nesta aula, os estudantes refletirão sobre o impacto das diversas tecnologias na cultura humana.

Objetivos	Refletir sobre o Impacto da Tecnologia na cultura Humana. Desenvolver uma visão crítica sobre o uso da Tecnologia.
Conteúdo	Conceito e Exemplos de Tecnologia. Impacto da Tecnologia na vida humana. Questões éticas relacionadas aos usos da Tecnologia.
Atividades	Refletindo sobre o Impacto da Tecnologia.
Resultados de Aprendizagem	C6. Criar, desenvolver, publicar e apresentar, de forma colaborativa, produtos (por exemplo, vídeos, podcasts, websites) usando recursos de tecnologia que demonstram e comunicam conceitos de currículo. C7. Colaborar com colegas, especialistas e outras pessoas usando práticas colaborativas, como programação em pares, trabalho em equipes de projeto e participação em atividades de aprendizado ativo em grupo. C8. Apresentar disposição necessária para colaboração: fornecer feedback útil, integrar feedback, compreender e aceitar múltiplas perspectivas, socialização. IC3. Discutir questões básicas relacionadas ao uso responsável de tecnologia e informação, e as consequências do uso inadequado. IC6. Identificar o impacto da tecnologia (por exemplo, redes sociais, cyberbullying, computação e comunicação móvel, tecnologias da Web, segurança cibernética e virtualização) na vida pessoal e na sociedade. IC10. Demonstrar conhecimento das mudanças nas tecnologias da informação ao longo do tempo e os efeitos que essas mudanças têm sobre a educação, local de trabalho e sociedade. IC14. Discutir como a distribuição desigual de recursos de computação em uma economia global levanta questões de equidade, acesso e poder.
Materiais	Folha de Atividades.

DESCRIÇÃO DAS ATIVIDADES

Nesta aula, os estudantes terão um momento de reflexão em grupo sobre o impacto da tecnologia na cultura humana. Inicialmente, o professor deve questionar os estudantes sobre "O que é Tecnologia?". Muitos estudantes irão dar exemplos de novas tecnologias. Então, o professor deve apresentar tecnologias mais antigas, como o fogo ou lápis, e perguntar se aquele objeto é uma tecnologia. Com isso, o professor deve pedir para que os alunos sistematizem o conceito de tecnologia.

Em seguida, o professor apresenta uma definição de tecnologia, passando pela ideia geral de ser um conjunto de técnicas, processos ou métodos utilizados na produção de algo, assim como, o desenvolvimento e utilização de ferramentas. Ao término da explicação, o professor pergunta novamente se os exemplos apresentados podem ser considerados tecnologias. Outro questionamento que deve ser feito aos estudantes é "Como essas tecnologias mais antigas modificaram o nosso modo de viver?". É fato que alguns destes aparatos facilitaram atividades humanas e, em consequência, modificaram a sociedade humana.

O professor então solicita que os estudantes, em grupos, respondam a um questionamento mais geral: "Qual o impacto da tecnologia na cultura humana?". Cada grupo deve ter um coordenador e um secretário. O professor deve explicar aos alunos a função de cada componente:

Função do Coordenador: Gerir o tempo e estimular todos os participantes a falarem sobre os questionamentos. O coordenador é responsável por não permitir respostas iguais, de modo que todos contribuam, além de garantir que o grupo mantenha o foco no tema da aula.

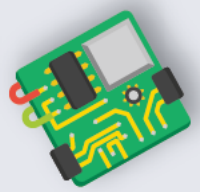
Função do Secretário: Tomar nota de todas as contribuições. Gerar um resumo ao término das contribuições. Ao final da atividade, cada secretário deve apresentar o resumo para toda a turma.

Função dos demais componentes: Fornecer respostas e exemplos para o questionamento dado pelo professor através da sua visão de mundo. Cada componente deve dar respostas e exemplos únicos (não dados por outros componentes do mesmo grupo).

Ao término da aula, o professor pode levantar algumas questões adicionais que não foram tratadas pelos grupos.

FOLHA DE ATIVIDADES

Refletindo sobre o Impacto da Tecnologia

Responda à seguinte questão: **"Qual o impacto da tecnologia na cultura humana?".** Analise quando este impacto é positivo e quando ele é negativo. Além disso, que soluções você poderia propor para aumentar este impacto, em caso positivo, ou diminuir, em caso negativo. Apresente, ao menos, três exemplos dos impactos a seguir.

- **Comportamentos sociais potencializados pela tecnologia**: as novas tecnologias trouxeram muitos avanços para a sociedade. Com isto, a tecnologia modifica o nosso jeito de viver para o bem e para o mal. Assim, alguns comportamentos não desejados podem ser potencializados através do uso incorreto da tecnologia.

- **Uso da informação:** A informação é algo muito importante atualmente. Sabemos que diversas empresas têm como núcleo do seu negócio uso de informações. Hoje todos nós podemos facilmente divulgar informações em sites e redes sociais. Em consequência, também recebemos informações de várias pessoas. Será que sabemos distinguir entre uma informação verídica ou falsa?

- **Uso de redes sociais**: As redes digitais diminuem distâncias entre pessoas e possibilitam que pessoas que talvez nunca se conhecessem possam compartilhar experiências de vida ou conhecer a cultura de outros lugares. Mas à medida que as redes aproximam o que está longe, podem distanciar o que está perto?

- **Computação móvel:** Ao longo da história, os computadores foram diminuindo de tamanho, através do estudo de novas tecnologias. A Computação móvel permite que levemos aparelhos a vários lugares como cinema, teatro, shows, viagens de férias e outros, mas até que ponto essa onipresença dos computadores pode ser positiva ou negativa?

- **Privacidade**: a facilidade quanto ao compartilhamento de informações trouxe também a quebra da privacidade. Muitas pessoas postam sobre suas vidas de forma excessiva. Além disso, algum aspecto da sua vida pode está online, em poucos segundos, sem sua autorização.

- **Segurança**: nossos computadores e celulares estão cheios de dados sobre nós: textos, fotos, vídeo, áudios e informações secretas, como senhas, por exemplo. Devemos tomar cuidado para que esses aparelhos não caiam em mãos erradas.

- **Acesso à tecnologia**: o mundo é muito grande e muito diversos. Você já deve ter ouvido que existem poucos com muitos e muitos com pouco. Este fato é real e se aplica a diversas coisas, como por exemplo a tecnologia. Já pensou que pode existir alguém no mundo que nunca teve acesso a um rádio, a uma televisão ou até mesmo a um celular?

AULA 2 – Minha nova fábrica de chocolate

Nesta aula, os estudantes aprenderão com atividades práticas e desplugadas conceitos relacionados ao design de interfaces humano-máquina.

Objetivos	Analisar diversos problemas relacionados ao design de Interfaces.
	Propor melhorias de design para problemas encontrados.
	Entender representações simbólicas (ícones).
	Implementar soluções conforme efeitos de transferência e estereótipos populacionais.
Conteúdo	Design de interfaces.
	Affordances.
	Mapeamento.
	Efeitos de transferência.
	Estereótipos populacionais.
	Ícones.
	Avaliação da interface do usuário.
Atividades	O problema das portas.
	O problema dos fogões.
	Projetando um novo sistema de comunicação.
	Organizando a fábrica.
	Identificando e criando ícones.
Resultados de Aprendizagem	PC2. Usar passos básicos na solução de problemas algorítmicos para projetar soluções (por exemplo, declaração e exploração de problemas, exemplos de instâncias, design, implementação de uma solução, teste, avaliação).
	C7. Colaborar com colegas, especialistas e outras pessoas usando práticas colaborativas, como programação em pares, trabalho em equipes de projeto e participação em atividades de aprendizado ativo em grupo.
	C8. Apresentar disposição necessária para colaboração: fornecer feedback útil, integrar feedback, compreender e aceitar múltiplas perspectivas, socialização.
Materiais	Folha de Atividades.

DESCRIÇÃO DAS ATIVIDADES

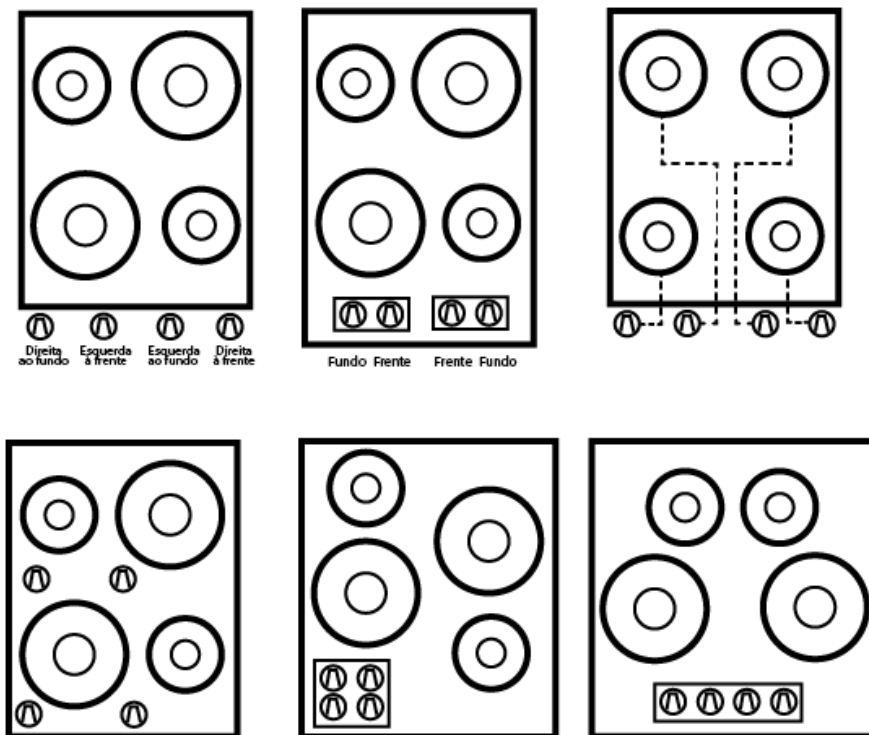
Nesta aula os estudantes aprenderão conceitos relacionados ao design de interfaces humano-máquina através atividades desplugadas.

1ª Parte: O professor deve iniciar a aula contextualizando os estudantes sobre a seguinte estória: "Uma fábrica de chocolate é administrada por uma raça de seres elfos, os Oompa-Loompas. Estes elfos têm péssima memória e não possuem nenhuma linguagem escrita. Por isso, eles têm dificuldade em lembrar o que fazer para administrar a fábrica, gerando vários acidentes durante a produção do chocolate. Por este motivo, está sendo projetada uma nova fábrica, que deverá ser muito adequada às necessidades deles". Os alunos, então, resolvem a primeira Folha de Atividades sobre o problema das portas. Depois de preencherem as suas próprias folhas, os estudantes devem discutir sobre as vantagens de cada tipo de porta. Por exemplo, a percepção sobre como ela funciona, ou como seria adequado usá-la ao carregar um balde de chocolate quente. Por fim, eles devem decidir que tipos de portas e alças devem ser usadas na fábrica.

Após a atividade, o professor deve discutir com os alunos, utilizando como base os comentários na tabela a seguir.

Porta simples: Não consigo ver como abrir isso, exceto que, como não tem alavanca, deve ser necessário empurrar, em vez de puxar.	**Porta com etiqueta**: A etiqueta é como um pequeno manual do usuário. Mas uma porta precisa de um manual do usuário? E Oompa-Loompas não sabem ler!
Porta com dobradiça: Pelo menos você pode ver de que lado ela se abre.	**Porta com barra**: É claro que você deve empurrar a barra, mas de que lado?
Porta com alavanca: Alavancas como estas geralmente são para puxar ou deslizar.	**Porta com maçaneta**: A maçaneta mostra o que pegar, mas não se você deve empurrar ou puxar; ela provavelmente não desliza.
Porta com painel: É claro que você deve empurrar. O que mais você poderia fazer?	**Porta de vidro**: a pequena barra vertical deste lado indica para puxar; a barra horizontal mais longa indica para empurrar.
Porta deslizante: Esta é apenas para deslizar.	

O conceito-chave aqui é o que chamamos de *affordances* de um objeto, que são suas características visíveis – fundamentais e percebidas – cuja aparência indica como o objeto deve ser usado. Por exemplo, é claro pela aparência que as cadeiras são para sentar, as mesas são para colocar as coisas, os botões são para girar, os compartimentos são para inserir coisas, os botões são para empurrar.

2ª Parte: No segundo momento, os estudantes devem avaliar e propor novos modelos para os fogões da fábrica. O conceito principal aqui é o *mapeamento* dos controles para seus resultados no mundo real. O mapeamento natural, que tira proveito das analogias físicas e dos padrões culturais, leva ao entendimento imediato. A figura seguinte apresenta alguns layouts possíveis. Alguns necessitam de descrição pois podem ter várias combinações, outros tiram proveito da disposição física.

3ª Parte: Os estudantes devem executar a terceira atividade, agora sobre um projeto de Comunicação na fábrica para indicar quando as esteiras devem parar, reduzir a velocidade ou iniciar. No entanto, deve-se levar em conta o padrão adotado no mundo dos Oompa-Loompas. No mundo deles, amarelo significa parar, vermelho significa seguir, e as luzes ficam verdes para avisar às pessoas que em breve terão de parar. Como isso afeta as coisas?
Os conceitos-chave aqui são os *efeitos de transferência* – as pessoas transferem seu aprendizado e expectativas de objetos anteriores para situações novas, mas semelhantes – e *estereótipos populacionais* - diferentes populações aprendem certos comportamentos e esperam que as coisas funcionem de uma certa maneira. Outros exemplos são os interruptores, teclados, números de telefones e datas em diversos países.

4ª Parte: Por fim, os estudantes devem identificar quais as mensagens passadas por cada ícone na Folha de Atividades (última parte). Não há respostas corretas para os ícones, a ideia é gerar discussão. Logo após, cada grupo deve implementar os seus próprios ícones, conforme os temas disponibilizados pelo professor. O professor pode utilizar: proibido mexer no celular, proibido andar descalços, envio de uma mensagem de áudio, recebimento de uma mensagem com imagem e outros. Ao término da atividade, os estudantes devem compartilhar seus ícones com os colegas de modo a obter *feedback*.

FOLHA DE ATIVIDADES

O problema das portas

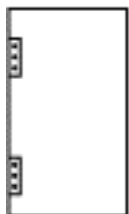
Analise cada tipo de porta e marque o seu modo de abrir, de acordo com a figura apresentada.
Você pode marcar mais de uma opção, se fizer sentido.

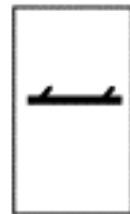
Porta Simples

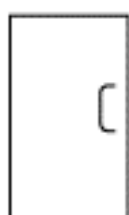
- ☐ Empurrar
- ☐ Puxar
- ☐ Deslizar
- ☐ Lado Esquerdo
- ☐ Lado Direito

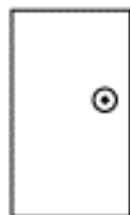
Porta com Etiqueta

- ☐ Empurrar
- ☐ Puxar
- ☐ Deslizar
- ☐ Lado Esquerdo
- ☐ Lado Direito

Porta com Dobradiça

- ☐ Empurrar
- ☐ Puxar
- ☐ Deslizar
- ☐ Lado Esquerdo
- ☐ Lado Direito

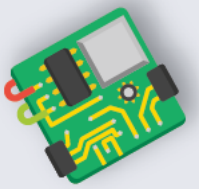
Porta com Barra

- ☐ Empurrar
- ☐ Puxar
- ☐ Deslizar
- ☐ Lado Esquerdo
- ☐ Lado Direito

Porta com Alavanca

- ☐ Empurrar
- ☐ Puxar
- ☐ Deslizar
- ☐ Lado Esquerdo
- ☐ Lado Direito

Porta com Maçaneta

- ☐ Empurrar
- ☐ Puxar
- ☐ Deslizar
- ☐ Lado Esquerdo
- ☐ Lado Direito

Porta com Painel

- ☐ Empurrar
- ☐ Puxar
- ☐ Deslizar
- ☐ Lado Esquerdo
- ☐ Lado Direito

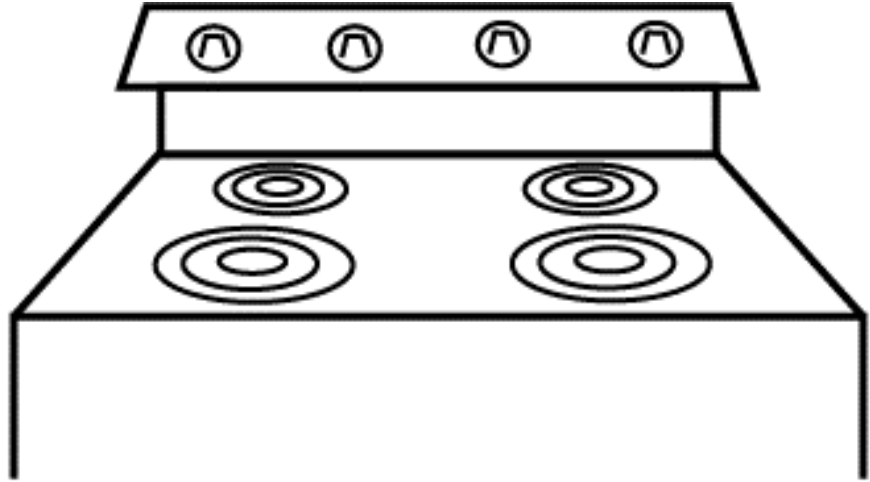
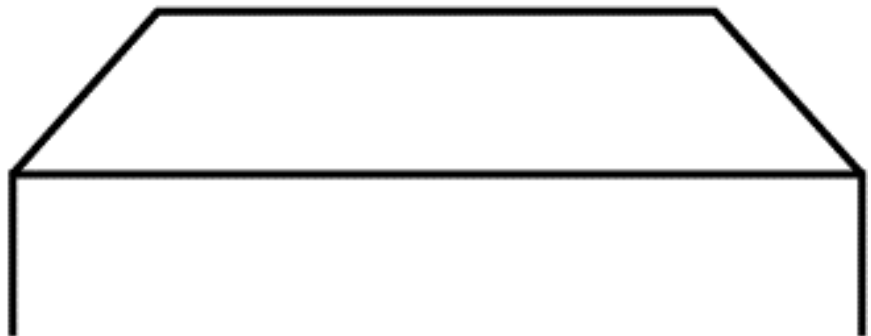
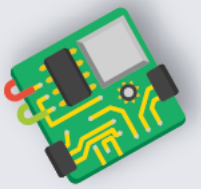
Porta de Vidro

- ☐ Empurrar
- ☐ Puxar
- ☐ Deslizar
- ☐ Lado Esquerdo
- ☐ Lado Direito

Porta Deslizante

- ☐ Empurrar
- ☐ Puxar
- ☐ Deslizar
- ☐ Lado Esquerdo
- ☐ Lado Direito

FOLHA DE ATIVIDADES

O problema dos fogões

Redesenhe o projeto do fogão para que ele fique mais fácil de utilizar.

FOLHA DE ATIVIDADES

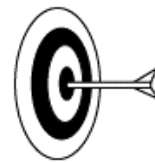
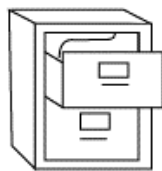
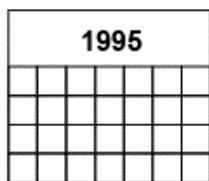
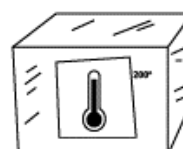
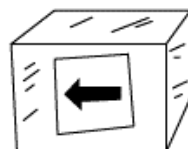
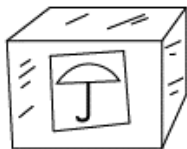
Projetando um novo sistema de comunicação

A fábrica está cheia de correias transportadoras carregando potes de chocolate. Essas correias transportadoras são controladas manualmente pelos Oompa-Loompas, sob instruções de uma sala de controle central. Na antiga fábrica, isso era feito com um sistema de voz: a voz da pessoa da sala de controle saía de um alto-falante pelos controles da correia transportadora. Mas a fábrica era barulhenta e difícil de ouvir. **Descreva um esquema que use sinais visuais em substituição aos alto-falantes.** *Uma possibilidade é utilizar semáforos. Pergunte ao professor como funcionam os Semáforos no mundo dos Oompa-Loompas.*

FOLHA DE ATIVIDADES

Identificando e criando ícones

Discuta, em grupo, para que servem esses ícones.

Em um Escritório

Em uma Caixa

Agora, em grupo, crie ícones para a atividade descrita no cartão entregue pelo professor.

AULA 3 – A matemática do movimento

Nesta aula, os estudantes investigarão como a modelagem computacional pode ser utilizada para entender ou descrever fenômenos da natureza, através de simulações. Os projetos desenvolvidos nesta aula tratam da ideia de descrever a queda livre como um movimento com aceleração constante, de modo similar às ideias de Galileu Galilei.

Objetivos	Analisar eventos da natureza a fim de encontrar um algoritmo que simule este mesmo evento.
	Descrever um modelo computacional para representação do movimento de queda livre.
	Entender como representações algébricas e suas respectivas adaptações para modelos discretos tornam possível a criação de simulações.
	Implementar animações que simulem eventos da natureza.
Conteúdo	Cinemática do movimento uniformemente variado.
	Comandos Scratch: Movimento, Eventos, Controle, Sensores, Operadores e Variáveis.
	Programação: Variáveis, Entrada e saída, Loops, Condicionais, Contadores, Expressões aritméticas.
Atividades	Projeto: A bola caindo.
	Projeto: A matemática do movimento.
Resultados de Aprendizagem	PC1. Descrever como uma simulação pode ser usada para resolver um problema.
	PC2. Usar passos básicos na solução de problemas algorítmicos para projetar soluções (por exemplo, declaração e exploração de problemas, exemplos de instâncias, design, implementação de uma solução, teste, avaliação).
	PC4. Definir um algoritmo como uma sequência de instruções que pode ser processada por um computador.
	PC5. Avaliar maneiras pelas quais diferentes algoritmos podem ser usados para resolver o mesmo problema.
	PC7. Descrever e analisar uma sequência de instruções que estão sendo seguidas (por exemplo, descrever o comportamento de um personagem em um videogame conforme orientado por regras e algoritmos).
	PC10. Interagir com modelos e simulações de conteúdo específico (por exemplo, ecossistemas, epidemias, dinâmica molecular) para apoiar a aprendizagem e pesquisa.

	PC11. Avaliar quais tipos de problemas podem ser resolvidos usando modelagem e simulação. PC12. Analisar o grau em que um modelo de computador representa com precisão o mundo real. PC15. Examinar as conexões entre os elementos da matemática e da ciência da computação, incluindo números binários, lógica, conjuntos e funções. PCP11. Implementar soluções de problemas usando uma linguagem de programação, incluindo: comportamento de looping, instruções condicionais, lógica, expressões, variáveis e funções.
Materiais	Scratch instalado ou online. projeto1_unidade1_aula3.sb3. Folha de Atividades.

DESCRIÇÃO DAS ATIVIDADES

Nesta aula, os estudantes investigarão como a modelagem computacional pode ser utilizada para entender ou descrever fenômenos da natureza, através de simulações. Os projetos desenvolvidos nesta aula tratam da ideia de descrever a queda livre como um movimento com aceleração constante, de modo similar às ideias de Galileu Galilei.

1ª Parte: O professor deve iniciar a aula indicando aos estudantes que eles devem fazer o primeiro exercício da Folha de Atividades. O professor deve reservar um tempo da aula para que os estudantes implementem o projeto previsto no Exercício 1 e respondam às questões do Exercício 2. O professor não deve oferecer explicações prévias sobre os termos queda livre, velocidade ou aceleração.

2ª Parte: O professor deve solicitar a alguns estudantes que apresentem suas animações, explicando o algoritmo implementado. Deve ser destacadas estratégias diferentes de implementação, caso ocorram. A partir das animações, o professor deve iniciar uma discussão em torno das questões previstas no Exercício 2. Ao final da discussão, a turma deve chegar a um consenso sobre a ideia de um movimento de queda livre e os conceitos de velocidade e aceleração.

3ª Parte: Os estudantes devem criar uma nova animação que represente a queda da bola, seguindo o algoritmo previsto no Exercício 3. O professor deve estipular um tempo da aula para que todos desenvolvam a animação.

4ª Parte: O professor pode convidar alguns estudantes para que apresentem a nova animação. Deve iniciar com a turma uma discussão sobre como este novo modelo de representação soa mais natural. O professor pode exemplificar as diferentes estratégias exibindo a animação das três bolinhas do projeto1_unidade1_aula3. O professor não deve apresentar detalhes de implementação, mas aproveitar a animação para discutir questões pertinentes sobre a representação da queda de uma bola.

5ª Parte: Os estudantes devem implementar a animação solicitada no Exercício 4 da Folha de Atividades. É importante que eles percebam por si sós as similaridades e divergências deste novo modelo em comparação ao modelo do Exercício 3. O professor deve evidenciar que, ao criar uma simulação no computador, é necessário adaptar equações de modo que o fenômeno que elas representam possa ser representado em um computador, que é uma máquina limitada.

FOLHA DE ATIVIDADES

A programação de computadores também pode ser empregada para a criação de modelos que representem o mundo real. As chamadas simulações utilizam a formalização matemática para construir modelos computacionais que imitam uma situação real. Um exemplo de utilização prática dos modelos computacionais é quando criamos jogos onde objetos caem, ou quando criamos animações que representem experimentos científicos.

Exercício 1 – A queda

Crie uma animação no Scratch onde uma bola deve "cair" no chão da tela. Ao criar sua animação, utilize um algoritmo que torne a queda da bola mais parecida com o modo como os objetos caem no mundo real.

Exercício 2 – Questões sobre a queda

Após finalizar a animação do Exercício 1, responda às questões a seguir:

1 – Ao jogar uma bola de uma determinada altura, ela mantém a mesma velocidade durante toda a queda?

____
____

2 – Os objetos em queda sempre aceleram? Em caso afirmativo, como?

____
____

3 – Como implementar uma simulação que represente a queda dos objetos? Descreva brevemente o algoritmo empregado.

____
____
____

4 – Na sua concepção, o que significa o termo **velocidade**?

____
____
____

5 – Na sua concepção, o que significa o termo **aceleração**?

____
____
____

Exercício 3 – Um modelo melhor para a queda

Na vida real, a queda é um movimento com aceleração constante. Esta ideia é uma das grandes descobertas do físico, matemático, astrônomo e filósofo Galileu Galilei.

Quando jogamos um objeto em repouso a partir de uma determinada altura, no primeiro instante, a velocidade é zero, mas, a cada instante que passa, há um aumento na velocidade. Do modo como os computadores representam este fenômeno, a cada repetição do programa, que corresponde a um "incremento de tempo", geramos um incremento constante na quantidade de pixels em que o objeto é deslocado na tela, o que corresponde, na prática, à velocidade. Assim, a **velocidade** é o incremento para a posição do objeto a cada instante e a **aceleração** é o incremento para a velocidade a cada instante. A queda respeita o que chamamos de movimento uniformemente variado, onde a velocidade aumenta e a aceleração é constante.

Recrie sua animação de bola caindo, empregando o algoritmo a seguir:

1. A velocidade inicial é zero.
2. A aceleração é um valor que pode ser alterado pelo usuário da simulação (utilize um controle deslizante para isso).
3. Repita os seguintes comandos até alcançar o chão:
   a. Ande o deslocamento especificado pela velocidade.
   b. A velocidade no instante atual = a velocidade no instante anterior + aceleração

Este modelo não é exatamente o modelo que representa a vida real, mas a queda seguindo esta representação é muito mais convincente!

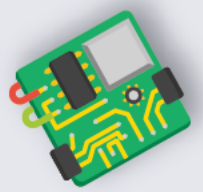
Exercício 4 – Queda livre

O conceito de queda livre é o movimento resultante unicamente da aceleração provocada pela gravidade, que é uma das forças fundamentais da natureza. A gravidade consiste em uma atração gravitacional da Terra, que dá peso aos objetos. Por isso é que as coisas caem ao chão quando são soltas!

Na prática, quando jogamos algo do alto de um prédio, além da força gravitacional, há também a resistência do ar interferindo na queda. Para efeitos de aprendizagem, podemos considerar que a resistência do ar é desprezível (ou fingir que estamos jogando um objeto do alto de um prédio e estamos no vácuo total)! Considerando apenas a gravitação atuando sobre o objeto caindo, a fórmula que descreve a queda livre é:

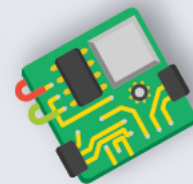
$$y = y_0 + v_0 t - \frac{1}{2} g t^2$$

y é a posição do objeto no instante t;
y_0 é a posição inicial do objeto;
v_0 é a velocidade inicial do objeto, se o objeto estava inicialmente em repouso então $v_0 = 0$;
g é a aceleração da gravidade, que, para os casos próximos à superfície do planeta Terra, pode ser aproximada para $g = 9{,}8\, m/s^2$.

Adapte este modelo matemático da queda livre de modo que ele possa ser representando em uma animação no Scratch, que explique ao usuário os principais conceitos da queda livre.

AULA 4 – Modelo Atômico de Bohr

Nesta aula, os estudantes devem implementar uma simulação que represente o modelo atômico de Bohr para um elemento químico de sua escolha.

Objetivos	Analisar eventos da natureza a fim de encontrar um algoritmo que simule este mesmo evento. Descrever um modelo computacional para a representação do movimento circular. Implementar animações que simulem o modelo atômico.
Conteúdo	Modelo atômico de Bohr. Comandos Scratch: Movimento, Eventos, Controle, Operadores e Variáveis. Programação: Variáveis, Entrada e saída, Loops, Condicionais, Contadores, Expressões aritméticas.
Atividades	Projeto: Simulação do modelo atômico de Bohr.
Resultados de Aprendizagem	PC1. Descrever como uma simulação pode ser usada para resolver um problema. PC2. Usar passos básicos na solução de problemas algorítmicos para projetar soluções (por exemplo, declaração e exploração de problemas, exemplos de instâncias, design, implementação de uma solução, teste, avaliação). PC4. Definir um algoritmo como uma sequência de instruções que pode ser processada por um computador. PC5. Avaliar maneiras pelas quais diferentes algoritmos podem ser usados para resolver o mesmo problema. PC7. Descrever e analisar uma sequência de instruções que estão sendo seguidas (por exemplo, descrever o comportamento de um personagem em um videogame conforme orientado por regras e algoritmos). PC10. Interagir com modelos e simulações de conteúdo específico (por exemplo, ecossistemas, epidemias, dinâmica molecular) para apoiar a aprendizagem e pesquisa. PC11. Avaliar quais tipos de problemas podem ser resolvidos usando modelagem e simulação. PC12. Analisar o grau em que um modelo de computador representa com precisão o mundo real. PC15. Examinar as conexões entre os elementos da matemática e da ciência da computação, incluindo números binários, lógica, conjuntos e funções. PCP11. Implementar soluções de problemas usando uma linguagem de programação,

	incluindo: comportamento de looping, instruções condicionais, lógica, expressões, variáveis e funções.
Materiais	Scratch instalado ou online. projeto1_unidade1_aula4.sb3. projeto2_unidade1_aula4.sb3. Tópicos Relevantes. Folha de Atividades.

DESCRIÇÃO DAS ATIVIDADES

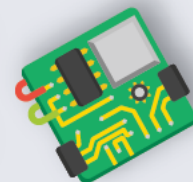
Nesta aula, os estudantes devem implementar uma simulação que represente o modelo atômico de Bohr para um elemento químico de sua escolha.

1ª Parte: O professor deve iniciar a aula com a leitura do texto sobre modelo atômico de Bohr na seção Tópicos Relevantes. O professor deve mediar a leitura, convidando alguns estudantes para ler determinados trechos. Também é importante que a discussão esteja pautada em conhecimentos prévios dos estudantes.

2ª Parte: O professor deve abrir e executar o programa projeto2_unidade1_aula4.sb3, que simula o modelo atômico de Bohr para o átomo de oxigênio. Neste momento, devem ser discutidas estratégias de implementação desta simulação. O professor deve perguntar a turma:

- Quais os atores?
- Que algoritmo pode ser utilizado para implementar o movimento circular dos elétrons?
- Como implementar um átomo com mais camadas energéticas?

O professor deve iniciar uma discussão sobre como implementar o movimento circular dos elétrons. As simulações computacionais costumam ser implementadas através da interpretação de fórmulas matemáticas. Então, deve apresentar as equações paramétricas para o círculo, anotando-as no quadro:

$$x = r \times \cos t$$
$$y = r \times \operatorname{sen} t$$

O professor deve explicar que estas duas equações servem para definir a posição de uma partícula. A posição ***x*** é expresso em função do cosseno e a posição ***y*** é expressa em função do seno. A variável ***r*** é o raio do círculo enquanto a variável ***t*** é uma variável que deve ser incrementada constantemente, como o tempo. Caso a turma não tenha nenhum conhecimento sobre seno e cosseno, o professor pode oferecer uma breve explicação sobre estas duas funções.

O professor deve abrir o programa projeto1_unidade1_aula4.sb3 e executá-lo. Então, junto com a turma devem interpretar o movimento das bolinhas e como este movimento está implementado:

- O ator Bola 1 se movimenta apenas na horizontal, pois seu programa implementa apenas a equação paramétrica para a posição ***x***, enquanto a posição ***y*** permanece com o valor constante zero;
- O ator Bola 2 se movimenta apenas na vertical, pois seu programa implementa apenas a equação paramétrica para a posição ***y***, enquanto a posição ***x*** permanece com o valor constante zero;
- O ator Bola 3 tem um movimento circular, pois seu programa implementa as equações paramétricas tanto para ***x*** como para ***y***.

Ao falar de cada ator, é importante que o professor mostre o programa implementado e discuta como a fórmula paramétrica foi implementada.

O professor deve retomar a animação do átomo de oxigênio, enfatizando características importantes:

- Cada camada é um círculo com raio de tamanho diferente.
- Os elétrons da camada K estão se movimentando no sentido horário, enquanto os elétrons da camada L movimentam-se no sentido anti-horário. Isso pode ser implementado adicionando uma variável de velocidade angular (ou frequência angular), que define o sentido do movimento e também a sua velocidade:

$$x = r \times \cos(a \times t)$$
$$y = r \times \text{sen}(a \times t)$$

Se a é um número negativo então o sentido do movimento é alterado! No exemplo abaixo, a velocidade é -3 e o ator terá seu movimento no sentido horário. Se a velocidade fosse 3, então o movimento circular ocorre no sentido anti-horário.

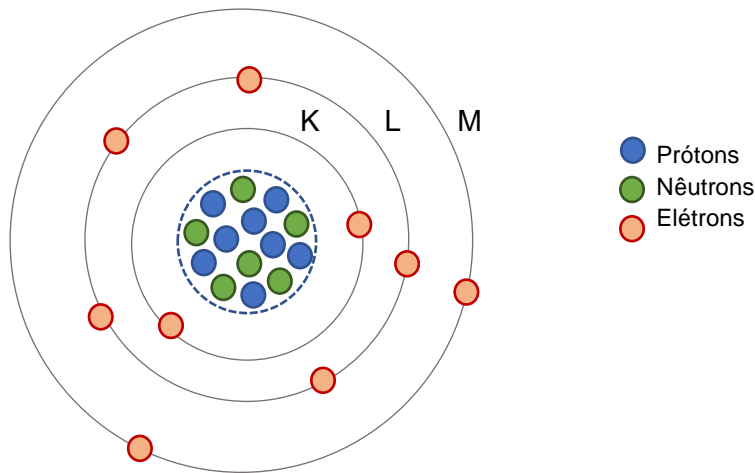
- Para fazer com que cada ator elétron mantenha uma certa distância em relação ao outro, é necessário adicionar uma constante de deslocamento. Por exemplo, na simulação do átomo de oxigênio a camada L possui 6 elétrons equidistantes. Como o círculo possui 360º, cada elétron deve ser posicionado com uma distância dos elétrons vizinhos de 60º, pois 360º/6 = 60º. Assim, o deslocamento do primeiro elétron é 0, do segundo é 60º, do terceiro é 120º, do quarto 180º, do quinto é 240º e do sexto elétron é 300º.

$$x = r \times \cos((a \times t) + d)$$
$$y = r \times \text{sen}((a \times t) + d)$$

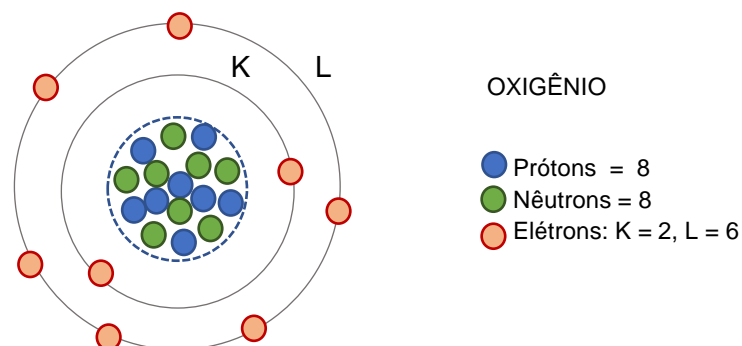
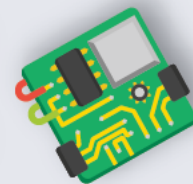
3ª Parte: Os estudantes devem escolher um elemento químico da tabela periódica e implementar uma simulação para seu átomo. Nesta etapa da aula, os estudantes devem seguir o guia previsto na Folha de Atividades. O professor deve mediar o processo para que cada estudante escolha um elemento químico diferente. O professor também deve estimular um tempo da aula para que todos implementem suas simulações e oferecer acompanhamento individual àqueles que apresentarem dúvidas.

4ª Parte: Ao final da aula, os estudantes devem apresentar suas simulações. Durante as apresentações, os estudantes devem explicar as características do elemento químico escolhido e descrever a estratégia de implementação empregada na simulação. Caso não haja tempo suficiente para que todos aproveitem, o professor deve selecionar alguns estudantes para a apresentação.

TÓPICOS RELEVANTES

Átomo é uma unidade básica de matéria, que é composto por **prótons**, **nêutrons** e **elétrons**. Um próton é uma partícula de carga elétrica positiva, enquanto um elétron é uma partícula de carga elétrica negativa. O nêutron não possui carga elétrica. No modelo atômico proposto pelo físico dinamarquês Niels Bohr, o átomo possui um núcleo pequeno e carregado positivamente cercado por elétrons em órbita circular, a exemplo do átomo da figura abaixo.

Algumas características do **modelo atômico de Bohr**:

- Existem sete camadas eletrônicas ao redor do núcleo e os elétrons estão distribuídos entre elas;
- As camadas são representadas pelas letras maiúsculas: K, L, M, N, O, P e Q. Existem sete níveis de energia, de modo que, à medida que as camadas se afastam do núcleo, aumenta a energia dos elétrons nelas localizados;
- Os elétrons seguem um movimento circular ao redor do núcleo do átomo.

Cada átomo possui uma quantidade específica de prótons, nêutrons, além dos elétrons de cada camada. À quantidade de camadas e de elétrons em cada camada do átomo, damos o nome de distribuição energética. Por exemplo: o átomo de oxigênio, que é um elemento que ocupa a posição 8 da tabela periódica, possui 8 prótons, 8 nêutrons e 8 elétrons.

O átomo de oxigênio possui duas camadas energéticas: a camada K possui 2 elétrons e a camada L possui 6 elétrons.

FOLHA DE ATIVIDADES

Nesta unidade, estamos aprendendo como a programação de computadores pode ser empregada para a criação de simulações. O átomo é um tema importante para o estudo da ciência, especialmente da química e, por isso, simulações de modelos atômicos são importantes ferramentas de estudos.

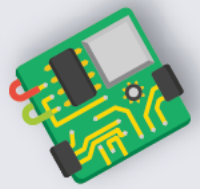
Exercício 1 – Escolha seu elemento químico

Escolha um dos elementos químicos da tabela periódica e responda às perguntas a seguir:

1 – Qual o elemento químico escolhido e seu número na tabela periódica?

____

2 – Quantos nêutrons o átomo do elemento escolhido possui em seu núcleo?

____

3 – Quantos prótons o átomo do elemento escolhido possui em seu núcleo?

____

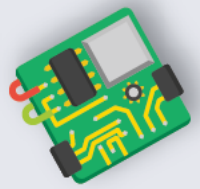
3 – Quantos elétrons o átomo do elemento escolhido possui em seu núcleo?

____

4 – Quantas camadas energéticas o átomo do elemento escolhido possui?

____

5 – Preencha a quantidade de elétrons em cada camada energética do átomo do elemento escolhido, deixando em branco as camadas que não existirem no átomo:

Camada energética	Quantidade de elétrons
K	
L	
M	
N	
O	
P	
Q	

Exercício 2 – Simule seu elemento

Crie uma animação no Scratch que represente o modelo atômico do elemento químico escolhido por você no Exercício 1.

Dica: As equações paramétricas para o círculo podem ser descrita como:

$$x = r \times \cos((a \times t) + d) \qquad y = r \times \operatorname{sen}((a \times t) + d)$$

Onde ***r*** é o raio do círculo, ***a*** é a aceleração da partícula, ***t*** é uma variável que é incrementada com o tempo e ***d*** é o deslocamento em relação à origem (0º) do movimento circular.

AULA 5 – Sistema Solar

Nesta aula, os estudantes devem implementar uma simulação que represente o movimento dos planetas no Sistema Solar. Os estudantes trabalharão com o movimento elíptico dos planetas em relação ao Sol.

Objetivos	Analisar eventos da natureza a fim de encontrar um algoritmo que simule este mesmo evento. Descrever um modelo computacional para representação do movimento elíptico. Implementar animações que simulem o movimento dos planetas no Sistema Solar
Conteúdo	Sistema Solar. Movimento elíptico. Comandos Scratch: Movimento, Eventos, Controle, Operadores e Variáveis. Programação: Variáveis, Entrada e saída, Loops, Condicionais, Contadores, Expressões aritméticas.
Atividades	Projeto: Simulação do Sistema Solar.
Resultados de Aprendizagem	PC1. Descrever como uma simulação pode ser usada para resolver um problema. PC2. Usar passos básicos na solução de problemas algorítmicos para projetar soluções (por exemplo, declaração e exploração de problemas, exemplos de instâncias, design, implementação de uma solução, teste, avaliação). PC4. Definir um algoritmo como uma sequência de instruções que pode ser processada por um computador. PC5. Avaliar maneiras pelas quais diferentes algoritmos podem ser usados para resolver o mesmo problema. PC7. Descrever e analisar uma sequência de instruções que estão sendo seguidas (por exemplo, descrever o comportamento de um personagem em um videogame conforme orientado por regras e algoritmos). PC10. Interagir com modelos e simulações de conteúdo específico (por exemplo, ecossistemas, epidemias, dinâmica molecular) para apoiar a aprendizagem e pesquisa. PC11. Avaliar quais tipos de problemas podem ser resolvidos usando modelagem e simulação. PC12. Analisar o grau em que um modelo de computador representa com precisão o mundo real. PC15. Examinar as conexões entre os elementos da matemática e da ciência da computação, incluindo números binários, lógica, conjuntos e funções.

	PCP11. Implementar soluções de problemas usando uma linguagem de programação, incluindo: comportamento de looping, instruções condicionais, lógica, expressões, variáveis e funções.
Materiais	Scratch instalado ou online.
	Vídeo “Primeira Lei de Kepler: Lei das Órbitas Elípticas (Astronomia)” disponível em: https://www.youtube.com/watch?v=g1b8zZ3LZhY.
	Tópicos Relevantes.
	Folha de Atividades.

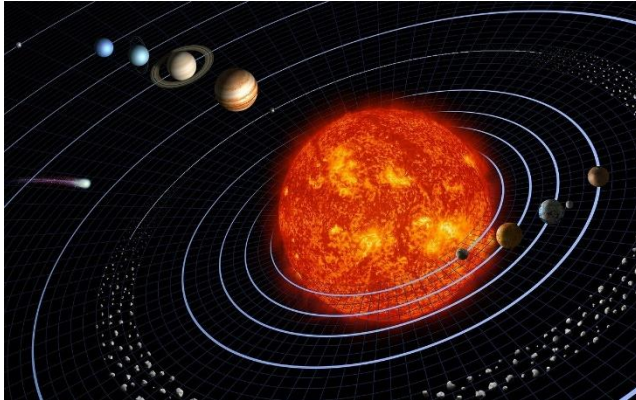
DESCRIÇÃO DAS ATIVIDADES

Nesta aula, os estudantes devem implementar uma simulação que represente o Sistema Solar. Os estudantes trabalharão com o movimento elíptico dos planetas em relação ao Sol.

1ª Parte: O professor deve iniciar a aula com a exibição do vídeo intitulado “Primeira Lei de Kepler: Lei das Órbitas Elípticas (Astronomia)”. Após a exibição do vídeo, deve ser iniciada a leitura do texto sobre Sistema Solar na seção Tópicos Relevantes. O professor deve mediar a leitura, convidando alguns estudantes para ler determinados trechos. Também é importante que a discussão esteja pautada em conhecimentos prévios dos estudantes.

2ª Parte: Os estudantes devem se dividir em duplas e implementar uma simulação do Sistema Solar. Os detalhes do projeto a ser implementado estão descritos na Folha de Atividades. Na descrição do projeto, há uma fórmula que descreve as coordenadas no plano cartesiano para uma elipse. As duplas devem interpretar e adaptar está fórmula para um modelo computacional viável no Scratch. O professor deve oferecer ajuda às duplas, quando solicitado.

3ª Parte: Ao final da aula, os estudantes devem apresentar suas simulações. Durante as apresentações, os estudantes devem explicar a estratégia de implementação empregada. Caso não haja tempo suficiente para que todos apresentem, o professor deve selecionar alguns grupos para fazê-lo.

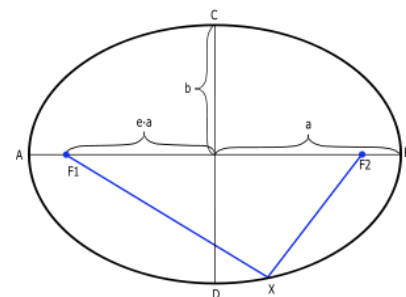
TÓPICOS RELEVANTES

O **Sistema Solar** é formado pelos planetas, satélites naturais e corpos celestes que estão sob domínio gravitacional do Sol.

Assim, o Sol é a estrela central do Sistema Solar, além de ser muito grande em relação a todos os outros elementos do sistema. Em nível de comparação, o volume do Sol, ou seja, a quantidade de espaço que ele ocupa, é 1.300.000 vezes maior do que a Terra.

O Sistema Solar possui 8 planetas conhecidos classificados de acordo com a distância em relação ao Sol: **Mercúrio**, **Vênus**, **Terra**, **Marte**, **Júpiter**, **Saturno**, **Urano** e **Netuno**. Os quatro planetas mais próximos do Sol são chamados planetas telúricos e têm como características comuns a presença de crostas sólidas e núcleos com elevada porcentagem de ferro. Os quatro planetas mais afastados são chamados planetas gigantes, pois possuem dimensões consideravelmente maiores que os planetas telúricos.

Além dos planetas conhecidos, há outros 5 planetas classificados como planetas anões: **Ceres**, **Plutão**, **Haumea**, **Makemake** e **Éris**.

Todos os planetas e demais corpos do Sistema Solar descrevem uma **órbita elíptica** ao redor do Sol. Uma elipse, por sua vez, é uma figura geométrica como ilustrada no exemplo abaixo. Este movimento orbital é também chamado de movimento de translação. Além disso, o tempo que cada planeta leva para estabelecer uma volta completa em torno do Sol varia. Na Terra, é de um ano. Já em Júpiter, dura em torno de 12 anos terrestres.

Uma elipse no plano cartesiano possui dois focos F1 e F2. O segmento de reta que passa por esses dois focos é chamado de eixo maior. A elipse também possui um eixo menor, que é perpendicular ao eixo maior. Quanto mais distantes estão os focos de uma elipse, maior é a sua **excentricidade** e, quando os focos estão mais próximos, a elipse se parece mais com um círculo.

Fonte: Wikipédia

Para a maior parte dos planetas, a elipse orbital é pouco excêntrica e a trajetória é praticamente circular. Nas órbitas do Sistema Solar, o Sol se localiza em um dos focos da elipse, fazendo com que haja um ponto onde ocorre a máxima aproximação dos corpos em relação ao Sol e um ponto oposto, onde ocorre a máxima distância em relação ao Sol.

Os componentes do Sistema Solar giram ao redor de seu próprio eixo imaginário, estabelecendo um movimento de rotação, que pode ser no sentido horário ou anti-horário. Vênus e Urano exercem a rotação no sentido anti-horário, enquanto todos os outros planetas o fazem no sentido horário.

FOLHA DE ATIVIDADES

Nesta unidade, estamos aprendendo como a programação de computadores pode ser empregada para a criação de simulações. O Sistema Solar é um tema importante para o estudo da ciência e da astronomia, por isso, simulações do comportamento dos planetas são importantes ferramentas de estudos.

Exercício 1 – Sistema Solar

Implemente no Scratch uma simulação para o Sistema Solar. Sua simulação deve representar os 8 planetas conhecidos Mercúrio, Vênus, Terra, Marte, Júpiter, Saturno, Urano e Netuno. Para que sua simulação fique muito legal, seguem algumas dicas:

1ª Dica: Pesquise as proporções dos planetas. No Sistema Solar o Sol possui um volume muito maior do que todos os outros corpos e os planetas também possuem diferentes tamanhos. Faça uma representação estabelecendo os tamanhos dos atores de sua animação de modo que eles se pareçam com a proporção original.

2ª Dica: A órbita dos planetas é elíptica e a posição do Sol é um dos focos destas elipses. A fórmula paramétrica utilizada na aula passada para descrever uma trajetória circular pode ser adaptada para que o movimento do ator seja elíptico. Uma elipse cujo eixo maior está no eixo x pode ser representada pelas seguintes equações:

Equações para o círculo:

$x = r \times \cos(velocidade \times t)$

$y = r \times \text{sen}(velocidade \times t)$

Equações para a elipse:

$x = a \times \cos(velocidade \times t)$

$y = b \times \text{sen}(velocidade \times t)$

Onde ***r*** é o raio do círculo, ***a*** é a metade do eixo maior (horizontal) da elipse e ***b*** é a metade do eixo menor (vertical) da elipse. Utilize os princípios desta fórmula para que os atores de sua simulação tenham uma trajetória elíptica.

3ª Dica: Cada planeta leva um determinado tempo para completar sua órbita! Pesquise e adicione diferentes valores de aceleração ao movimento de cada planeta.

4ª Dica: Faça com que sua simulação também represente o movimento de rotação, que é quando os planetas giram em torno de si mesmos. Lembre-se que cada planeta tem um tempo de rotação específico. Faça sua pesquisa e descubra proporções de tempos de rotação para os planetas.

AULA 6 - Fazendo seu Projeto de Ciências (parte 1)

Nesta aula, os estudantes devem exercitar suas habilidades sobre programação no Scratch para idealizar projetos de ciência.

Objetivos	Idealizar e planejar um projeto de ciência a ser desenvolvido no Scratch. Aplicar todos os conceitos aprendidos até então para criar um projeto de ciência.
Conteúdo	Conceitos específicos de ciências. Comandos Scratch: Movimento, Eventos, Controle, Sensores, Operadores e Variáveis. Programação: Variáveis, Entrada e saída, Loops, Condicionais, Contadores, Expressões aritméticas, Funções matemáticas.
Atividades	Idealizando meu projeto de ciências.
Resultados de Aprendizagem	PC1. Descrever como uma simulação pode ser usada para resolver um problema. PC2. Usar passos básicos na solução de problemas algorítmicos para projetar soluções (por exemplo, declaração e exploração de problemas, exemplos de instâncias, design, implementação de uma solução, teste, avaliação). PC4. Definir um algoritmo como uma sequência de instruções que pode ser processada por um computador. PC5. Avaliar maneiras pelas quais diferentes algoritmos podem ser usados para resolver o mesmo problema. PC7. Descrever e analisar uma sequência de instruções que estão sendo seguidas (por exemplo, descrever o comportamento de um personagem em um videogame conforme orientado por regras e algoritmos). PC10. Interagir com modelos e simulações de conteúdo específico (por exemplo, ecossistemas, epidemias, dinâmica molecular) para apoiar a aprendizagem e pesquisa. PC11. Avaliar quais tipos de problemas podem ser resolvidos usando modelagem e simulação. PC12. Analisar o grau em que um modelo de computador representa com precisão o mundo real. PC13. Usar abstração para decompor um problema em subproblemas. PC15. Examinar as conexões entre os elementos da matemática e da ciência da computação, incluindo números binários, lógica, conjuntos e funções. PCP11. Implementar soluções de problemas usando uma linguagem de programação,

	incluindo: comportamento de looping, instruções condicionais, lógica, expressões, variáveis e funções.
Materiais	Scratch instalado ou online. Folha de Atividades.

DESCRIÇÃO DAS ATIVIDADES

Nesta aula, os estudantes ficarão livres para criar seus próprios projetos de ciência em duplas. O professor deve pedir aos estudantes que escolham um fenômeno de ciências a ser simulado no computador com Scratch. O professor pode fornecer exemplos, conforme os conteúdos já aprendidos previamente pelos estudantes. O professor deve verificar os temas de modo a orientar os estudantes para que escolham temas muito complexos, que exigirão habilidades ainda não adquiridas, ou temas excessivamente simples, que não permitam o uso dos conceitos aprendidos até então.

Os estudantes, após a definição do tema, devem utilizar a Folha de Atividades para descrever seus projetos, além de assinalar quais os possíveis conteúdos que utilizarão. O professor deve atentar para que a descrição seja bastante detalhada e para que os estudantes não utilizem poucos conceitos de programação, pois a ideia é que eles explorem o máximo de conceitos trabalhados previamente.

FOLHA DE ATIVIDADES

Idealizando meu projeto de ciências

1 – Qual o fenômeno da natureza escolhido para a simulação?

____

2 - Descreva o seu projeto de forma detalhada. Além disso, você pode utilizar desenhos para exemplificar. Se você usar fórmulas matemáticas, coloque-as aqui também.

3 - Marque com um "X" os conceitos que utilizará:

[ ] Variáveis
[ ] Entrada
[ ] Saída
[ ] Condicionais
[ ] Loops
[ ] Contadores
[ ] Expressões aritméticas
[ ] Funções matemáticas
[ ] Ângulos

4 – Descreva aqui outros conceitos específicos de ciências que utilizará:

____
____
____
____
____
____

AULA 7 - Fazendo seu Projeto de Ciências (parte 2)

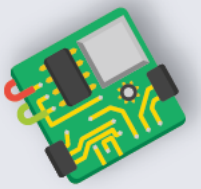
Nesta aula, os estudantes devem exercitar suas habilidades de programação em Scratch para implementar projetos de ciências idealizados por eles.

Objetivos	Desenvolver projetos de ciências com Scratch com base em um planejamento previamente elaborado.
	Aplicar todos os conceitos de programação aprendidos até então para criar um projeto de ciências.
Conteúdo	Conceitos específicos de ciências.
	Comandos Scratch: Movimento, Eventos, Controle, Sensores, Operadores e Variáveis.
	Programação: Variáveis, Entrada e saída, Loops, Condicionais, Contadores, Expressões aritméticas, Funções matemáticas.
Atividades	Desenvolvendo meu projeto de ciências.
Resultados de Aprendizagem	PC1. Descrever como uma simulação pode ser usada para resolver um problema.
	PC2. Usar passos básicos na solução de problemas algorítmicos para projetar soluções (por exemplo, declaração e exploração de problemas, exemplos de instâncias, design, implementação de uma solução, teste, avaliação).
	PC4. Definir um algoritmo como uma sequência de instruções que pode ser processada por um computador.
	PC5. Avaliar maneiras pelas quais diferentes algoritmos podem ser usados para resolver o mesmo problema.
	PC7. Descrever e analisar uma sequência de instruções que estão sendo seguidas (por exemplo, descrever o comportamento de um personagem em um videogame conforme orientado por regras e algoritmos).
	PC10. Interagir com modelos e simulações de conteúdo específico (por exemplo, ecossistemas, epidemias, dinâmica molecular) para apoiar a aprendizagem e pesquisa.
	PC11. Avaliar quais tipos de problemas podem ser resolvidos usando modelagem e simulação.
	PC12. Analisar o grau em que um modelo de computador representa com precisão o mundo real.
	PC13. Usar abstração para decompor um problema em subproblemas.
	PC15. Examinar as conexões entre os elementos da matemática e da ciência da computação, incluindo números binários, lógica, conjuntos e funções.

	PCP11. Implementar soluções de problemas usando uma linguagem de programação, incluindo: comportamento de looping, instruções condicionais, lógica, expressões, variáveis e funções.
Materiais	Scratch instalado ou online. Folha de Atividades.

DESCRIÇÃO DAS ATIVIDADES

Nesta aula, os estudantes irão implementar os projetos idealizados na aula anterior. O professor deve monitorar os alunos no sentido de gerenciar o tempo e de garantir que irão implementar o projeto idealizado, utilizando os conceitos de programação previamente aprendidos. O professor pode esclarecer eventuais dúvidas, mas não deve entregar respostas prontas, incentivando os estudantes a pensar em possíveis soluções.

Ao término do projeto, o professor deve pedir para que todas as duplas apresentem o tema que eles escolheram para simulação. Caso ainda haja tempo, as duplas podem interagir com os projetos das outras duplas. O ideal é que o professor possa garantir que esses projetos possam ser divulgados na Feira de Ciências da escola ou outro evento voltado para a comunidade escolar.

FOLHA DE ATIVIDADES

Desenvolvendo meu Projeto de Ciências

1 – Com base na Folha de Atividades da aula anterior, implemente no Scratch o seu projeto de Ciências.

Passo 1: Selecione as imagens que utilizará.
Passo 2: Desenhe as imagens que não encontrou.
Passo 3: Selecione áudios que utilizará;
Passo 4: Coloque as imagens no seu projeto Scratch.
Passo 5: Implemente sua simulação, programando cada ator por vez.

UNIDADE II

Manipulação de Imagens

AULA 1 – Pixel a Pixel

Nesta aula, os estudantes irão relembrar conceitos sobre o padrão de cores RGB, pixels, imagens digitais e manipularão os pixels através do JES de modo introdutório.

Objetivos	Compreender o padrão RGB de imagens digitais. Compreender as propriedades dos pixels. Manipular imagens através do JES com a linguagem Python.
Conteúdo	Imagens Digitais. Pixels. Padrão de cores RGB. Programação: variáveis, sequência, loops simples e funções de bibliotecas.
Atividades	Criando meus primeiros efeitos.
Resultados de Aprendizagem	PC2. Usar passos básicos na solução de problemas algorítmicos para projetar soluções (por exemplo, declaração e exploração de problemas, exemplos de instâncias, design, implementação de uma solução, teste, avaliação). PC5. Avaliar maneiras pelas quais diferentes algoritmos podem ser usados para resolver o mesmo problema. PC8. Representar dados de várias formas, incluindo texto, sons, imagens e números. PC12. Analisar o grau em que um modelo de computador representa com precisão o mundo real. PC15. Examinar as conexões entre os elementos da matemática e da ciência da computação, incluindo números binários, lógica, conjuntos e funções. PCP10. Demonstrar compreensão dos algoritmos e sua aplicação prática. PCP11. Implementar soluções de problemas usando uma linguagem de programação, incluindo: comportamento de looping, instruções condicionais, lógica, expressões, variáveis e funções.
Materiais	JES. Slide Editando Imagem no JES (slide1_unidade3_aula1.ppt).

DESCRIÇÃO DAS ATIVIDADES

Nesta aula, os estudantes irão relembrar conceitos sobre imagens digitais, pixels, padrão de cores RGB e aprenderão a manipular pixels através do JES, com Python. Inicialmente, o professor deve falar sobre imagens digitais, perguntando aos estudantes o que eles lembram.

O professor inicia a apresentação de slides relembrando o conceito de pixels e o padrão de cores RGB, a composição de uma imagem como uma matriz de pixels e suas coordenadas x e y.

Após esta apresentação inicial, o professor mostra o JES e a possibilidade de realizar modificações em imagens. Os estudantes devem abrir o JES e executar alguns comandos apresentados nos slides, editando imagens pixel a pixel. O professor deve sempre fazer o paralelo entre o GPBlocks e o JES com Python para que o estudante tenha um referencial. Esta aula é uma introdução ao JES e os alunos devem replicar os exemplos apresentados no slide. O professor pode aproveitar dúvidas específicas de estudantes para esclarecer para toda a turma, fazendo um paralelo entre a teoria e a prática de manipulação de imagens.

AULA 2 – Fazendo Seus Próprios Filtros

Nesta aula, os estudantes irão criar efeitos visuais que modificam os valores dos pixels originais, como matiz, utilizando loops e operações aritméticas sobre os pixels com o uso da linguagem Python.

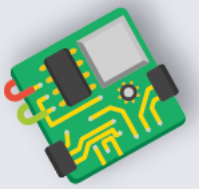
Objetivos	Compreender como operações matemáticas podem realizar efeitos de imagens. Manipular imagens com a ferramenta JES utilizando Python. Aprofundar conceitos sobre pixels e efeitos.
Conteúdo	Efeitos de imagem. Propriedades dos pixels. Funções do JES: getPixels, get/setRed, get/setGreen, get/setBlue. Programação: variáveis, expressões aritméticas, sequência, loops, listas e funções de bibliotecas.
Atividades	Criando efeitos de Matiz.
Resultados de Aprendizagem	PC2. Usar passos básicos na solução de problemas algorítmicos para projetar soluções (por exemplo, declaração e exploração de problemas, exemplos de instâncias, design, implementação de uma solução, teste, avaliação). PC5. Avaliar maneiras pelas quais diferentes algoritmos podem ser usados para resolver o mesmo problema. PC8. Representar dados de várias formas, incluindo texto, sons, imagens e números. PC15. Examinar as conexões entre os elementos da matemática e da ciência da computação, incluindo números binários, lógica, conjuntos e funções. PCP10. Demonstrar compreensão dos algoritmos e sua aplicação prática. PCP11. Implementar soluções de problemas usando uma linguagem de programação, incluindo: comportamento de looping, instruções condicionais, lógica, expressões, variáveis e funções.
Materiais	JES. Folha de Atividades.

DESCRIÇÃO DAS ATIVIDADES

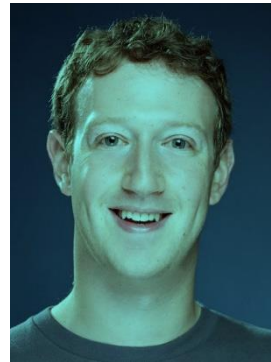
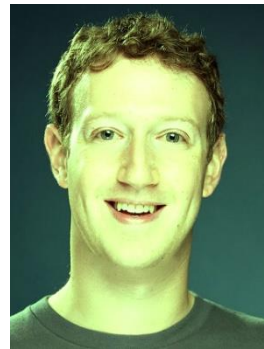
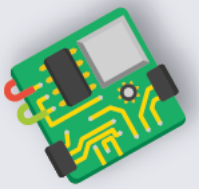
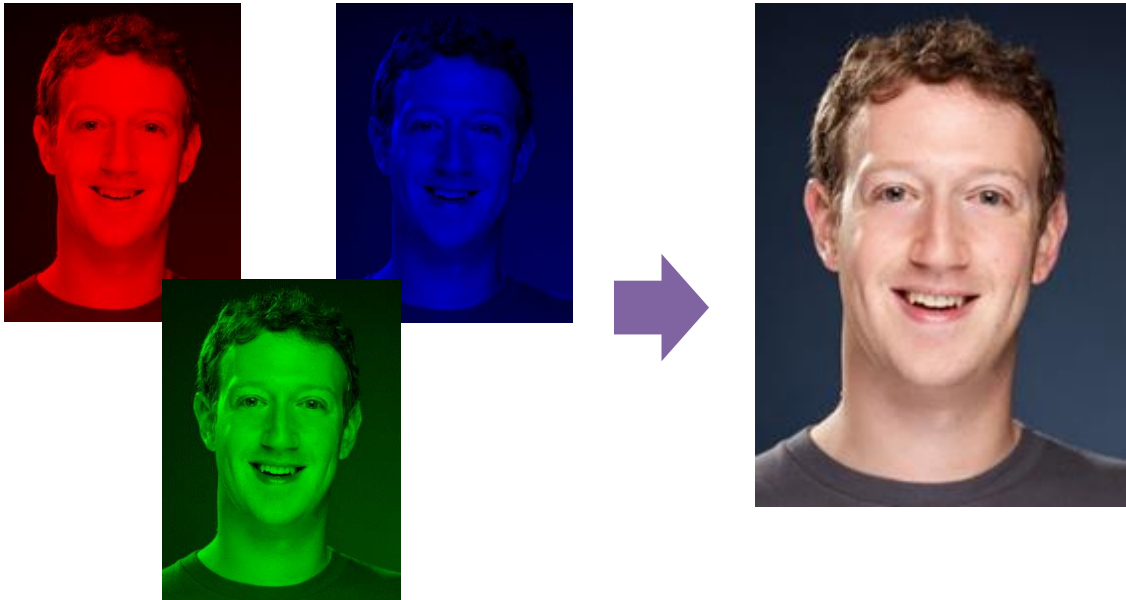
Nesta aula, os estudantes irão utilizar operações matemáticas para criar efeitos em imagens, utilizando a linguagem Python. O professor deve, juntamente com a turma, relembrar os conteúdos das atividades da aula anterior, assim como a sua relação com atividades vistas no oitavo ano com GP Blocks. Nas atividades desta aula, diferentemente da aula anterior, os alunos devem modificar as cores dos pixels de sem sobrescrever a imagem.

O professor deve relembrar os comandos *red of pixel*, *green of pixel* e *blue of pixel* do GP Blocks, e apresentar os comandos similares em Python para recuperar o valor das cores de um pixel (*getRed*, *getGreen* e *getBlue*), assim como os comandos para definir novos valores (*setRed*, *setGreen* e *setBlue*). Além dos comandos, os estudantes deverão utilizar operadores aritméticos, já vistos no Scratch e no GP Blocks. É interessante que o professor sempre reforce a similaridade entre as ferramentas.

Com o uso do projetor, o professor deve editar os pixels de uma imagem, utilizando loops e *getPixels*. Os estudantes devem, após a explicação do professor, fazer os desafios da Folha de Atividades, gerando efeitos e salvando cada programa. O professor deve ainda ensinar os estudantes a salvar as imagens geradas com a função *writePictureTo*. A figura a seguir ilustra um exemplo de efeito de matiz reduzindo o componente vermelho, usando loops e listas.

```
imagem = makePicture("C:/Usuario/Documentos/zuckberg.jpg")

arrayPixels = getPixels(imagem)
for p in arrayPixels:
  valorVermelho = getRed(p)
  setRed(p, valorVermelho*2)
```

FOLHA DE ATIVIDADES - Criando efeitos de Matiz

Exercite suas habilidades com pixels para a criação de efeitos variados.

- ✓ **Matiz Vermelha**. Reduza do valor vermelho (*red*) pela metade, em todos os pixels da imagem, e obtenha o seguinte efeito.

- ✓ **Matriz Verde.** Adicione 50% ao valor verde (*green*), em todos os pixels da imagem, e obtenha o seguinte efeito.

- ✓ **Matiz Azul.** Reduza 250 do valor azul (*blue*), em todos os pixels da imagem, e obtenha o seguinte efeito.

- ✓ **Desafio**. Você já sabe que as imagens RGB são formadas por 3 valores de cores básicas RGB: *Red, Green e Blue*. Experimente criar 3 imagens, cada uma com apenas uma dessas cores. Ao final, some os valores dos pixels das 3 imagens e obtenha a imagem original. Preparado?!

AULA 3 - Mais Efeitos Utilizando Matemática

Nesta aula, os estudantes irão aplicar os conhecimentos matemáticos para criar os efeitos de negativo, escala de cinza e espelhamento. Os estudantes irão desenvolver programas, utilizando funções, para criar os efeitos.

Objetivos	Aprofundar a aplicação de operações matemáticas para realizar efeitos de imagem.
	Manipular imagens com a ferramenta JES usando a linguagem Python.
	Compreender a estrutura e funcionamento de funções em Python.
Conteúdo	Efeitos de imagem.
	Propriedades dos pixels.
	Funções do JES: getPixels, get/setRed, get/setGreen, get/setBlue, getHeight, getWidth, get/setPixel.
	Programação: variáveis, expressões aritméticas, sequência, loops, listas e funções.
Atividades	Criando mais efeitos com operações matemáticas.
Resultados de Aprendizagem	PC2. Usar passos básicos na solução de problemas algorítmicos para projetar soluções (por exemplo, declaração e exploração de problemas, exemplos de instâncias, design, implementação de uma solução, teste, avaliação).
	PC5. Avaliar maneiras pelas quais diferentes algoritmos podem ser usados para resolver o mesmo problema.
	PC8. Representar dados de várias formas, incluindo texto, sons, imagens e números.
	PC15. Examinar as conexões entre os elementos da matemática e da ciência da computação, incluindo números binários, lógica, conjuntos e funções.
	PCP10. Demonstrar compreensão dos algoritmos e sua aplicação prática.
	PCP11. Implementar soluções de problemas usando uma linguagem de programação, incluindo: comportamento de looping, instruções condicionais, lógica, expressões, variáveis e funções.
Materiais	JES.
	Folha de Atividades.

DESCRIÇÃO DAS ATIVIDADES

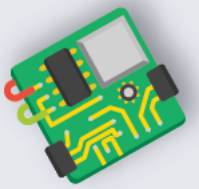
Nesta aula, os estudantes continuarão trabalhando com efeitos em imagens utilizando operações matemáticas. Como visto no ano anterior com GP Blocks, os estudantes podem criar efeitos de Escala de Cinza, Negativo e Espelhamento. Além disso, os estudantes terão um momento para exercitar sua criatividade juntando efeitos novos e já vistos.

O professor deve iniciar a aula, criando uma função como exemplo, utilizando o efeito de matiz da aula anterior. Em seguida, deve pedir aos estudantes que implementem os efeitos da Folha de Atividades, usando uma função para cada efeito.

Para criar um efeito Negativo, basta subtrair o valor de cada cor dos pixels do valor máximo de cor que um pixel pode assumir (255), como no código a seguir.

```
def negativo():
  pixels = imagem.getPixels()
  for p in pixels:
    novoRed = 255 - p.getRed();
    novoGreen = 255 - p.getGreen();
    novoBlue = 255 - p.getBlue();
    setColor(p, Color(novoRed,novoGreen,novoBlue))
```

Para criar um efeito Escala de Cinza, basta calcular o valor médias das cores red, green e blue de cada pixel e substituir estes valores pelo valor médio, como no código a seguir.

```
def escalacinza():
  pixels = imagem.getPixels()
  for p in pixels:
    soma = p.getRed()+p.getGreen()+p.getBlue();
    media = soma/3
    setColor(p, Color(media,media,media))
```

Para criar um efeito Espelhamento, basta usar os pixels da primeira metade da imagem e copiá-los para a segunda metade da imagem, percorrendo-a na direção contrária, como no código a seguir.

```
def espelharImagem():
  metade = int(getWidth(imagem)/2)
  for x in range(1, metade):
    for y in range(1, getHeight(imagem))
      p1 = getPixel(imagem, x+metade,y)
      p2 = getPixel(imagem, metade-x, y)
      setColor(p1,getColor(p2))
```

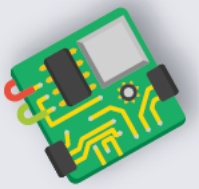
Além da função *getPixels*, os estudantes precisarão utilizar o *getHeight*, *getWidth* e *getPixel*. O professor deve demonstrar o funcionamento destas funções. Após a realização da tarefa, o professor seleciona três alunos para apresentação e explicação dos seus códigos.

Em seguida, os estudantes deverão exercitar sua criatividade juntando efeitos novos e já vistos anteriormente.

FOLHA DE ATIVIDADES - Criando mais efeitos com operações matemáticas

Aqui, você irá exercitar suas habilidades matemáticas e lógicas para a criação de efeitos mais avançados com Python. Não esqueça de utilizar funções.

- ✓ **Crie um efeito de Negativo**. Um efeito de negativo é criado através da subtração do valor original do pixel do valor máximo que uma cor do pixel pode receber: 255 – valor da cor.

- ✓ **Crie o efeito Escala de Cinza.** Um efeito escala de Cinza é criado utilizando a média das três cores RGB de cada pixel. Para cada pixel, é preciso somar os valores de cada cor e depois dividi-las por 3, calculando uma média. Esse novo valor deve ser colocado em cada cor RGB do mesmo pixel, ou seja, (red + green + blue) / 3

- **Crie o efeito de Espelhamento**. Um efeito de espelhamento é realizado copiando os pixels da primeira metade da imagem para a segunda metade, percorrendo-a na direção contrária. O loop padrão deve ser modificado para percorrer apenas metade do comprimento da imagem (*width* / 2). A cor de cada pixel percorrido é trocada pela cor do pixel na posição oposta. Por exemplo, em uma imagem de comprimento 100, o pixel 0,0 é oposto ao 99,0; o pixel 1,0 é oposto ao pixel 98,0. O pixel 20,10 é oposto ao pixel 79,10 e assim sucessivamente.

- **Novo Efeito**. Escolha uma personagem que você gosta. Misture dois ou mais efeitos ou realize modificações sobre os efeitos vistos para gerar novos efeitos. Escreva a seguir o que você fez para gerar este novo efeito e qual é o nome que você deu a ele.

____

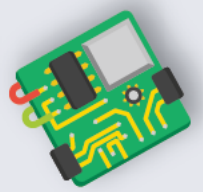
____

____

____

____

____

____

AULA 4 – Mais efeitos visuais

Nesta aula, os estudantes irão manipular imagens através de condicionais, criando recortes de imagens, efeito binário e adicionando bordas em imagens. Os estudantes devem usar funções com parâmetros para estipular as cores nos efeitos.

Objetivos	Aplicar operações matemáticas para realizar efeitos em imagens. Utilizar estruturas condicionais para obter efeitos específicos em imagens. Utilizar parâmetros em funções para criar efeitos sob medida. Manipular imagens em Python.
Conteúdo	Efeitos de imagem. Propriedades dos pixels. Funções do JES: getPixels, get/setRed, get/setGreen, get/setBlue, getHeight, getWidth, get/setPixel. Programação: variáveis, expressões aritméticas, condicionais, loops, listas e funções.
Atividades	Criando efeitos com condicionais.
Resultados de Aprendizagem	PC2. Usar passos básicos na solução de problemas algorítmicos para projetar soluções (por exemplo, declaração e exploração de problemas, exemplos de instâncias, design, implementação de uma solução, teste, avaliação). PC5. Avaliar maneiras pelas quais diferentes algoritmos podem ser usados para resolver o mesmo problema. PC8. Representar dados de várias formas, incluindo texto, sons, imagens e números. PC15. Examinar as conexões entre os elementos da matemática e da ciência da computação, incluindo números binários, lógica, conjuntos e funções. PCP10. Demonstrar compreensão dos algoritmos e sua aplicação prática. PCP11. Implementar soluções de problemas usando uma linguagem de programação, incluindo: comportamento de looping, instruções condicionais, lógica, expressões, variáveis e funções.
Materiais	JES. Folha de Atividades.

DESCRIÇÃO DAS ATIVIDADES

Nesta aula, os estudantes utilizarão condicionais para criar novos efeitos. Os estudantes irão aprender a aplicar estruturas condicionais em Python para criar o efeito Binário (Preto e Branco, Vermelho e Preto e outros); recortar imagens e adicionar bordas em imagens (modificando apenas as laterais). Para todos os efeitos, os estudantes devem criar funções que utilizem parâmetros como, por exemplo, escolha de cores para o efeito binário ou o tamanho da borda.

Para ilustrar o uso de parâmetros, o professor deve utilizar o efeito negativo da aula anterior, passando como parâmetro para a função, além da imagem, o valor a ser subtraído. O professor deve enfatizar a conceito de parâmetro em funções, assim como variar o valor passado, fazendo várias chamadas à função negativo.

```
def negativo():
  pixels = imagem.getPixels()
  for p in pixels:
    novoRed = 255 - getRed(p);
    novoGreen = 255 - getGreen(p);
    novoBlue = 255 - getBlue(p);
    setColor(p, Color(novoRed,novoGreen,novoBlue))
```

Ao término da explicação do professor sobre parâmetros em funções, os estudantes devem passar para a Folha de Atividades, realizando as atividades planejadas. A Folha de Atividades contém dois desafios adicionais para os alunos que fizerem as atividades mais rapidamente.

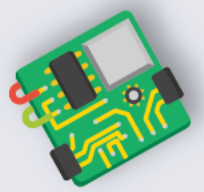
Como preparativo para a aula seguinte, o professor deve pedir aos estudantes que tragam imagens deles mesmos com um fundo padrão verde ou azul, e imagens adicionais de lugares que eles gostam, para a criação do efeito de *Chroma key*.

FOLHA DE ATIVIDADES

Criando efeitos com condicionais e funções com parâmetros

Chegou a hora de exercitar suas habilidades com Python e manipulação de imagens. Para isso, faça as seguintes atividades.

- ✓ **Crie um efeito binário**. Um efeito Binário é um efeito que transforma uma imagem com muitas cores em uma imagem com apenas duas cores. Para isso, você pode utilizar a comparação entre um limiar (valor fixo definido de intensidade da cor) e o valor da soma das cores RGB para decidir se pinta de uma cor ou de outra (*somaDasCores* < *limiar*). Crie uma função para fazer o efeito binário que receba como parâmetros as duas cores do efeito.

- ✓ **Crie um efeito de borda.** O efeito de borda já é um velho conhecido seu da primeira aula sobre imagens. Faça uma função para criar uma borda em uma imagem que utilize dois parâmetros: um para estipular a espessura da borda e outra para a cor da borda.

- ✓ **Crie o efeito de recorte**. Um efeito de recorte é um efeito realizado com mais de uma imagem. A ideia é copiar uma parte (retângulo) de uma imagem e adicioná-la em outra, sobrepondo a parte copiada sobre a segunda imagem. Veja este exemplo:

Faça uma função chamada recorte que receba dois parâmetros: a primeira imagem, que será copiada, e a segunda imagem, que receberá o recorte. Perceba que funções podem receber imagens como parâmetros.

Desafio 1. No efeito recorte, foram utilizadas duas imagens como parâmetros da função. Você consegue pensar em dois parâmetros adicionais para escolher o trecho da primeira imagem que será copiado? Que tal criar um código com estes novos parâmetros?

Desafio 2. No efeito de recorte, é possível deixar a imagem do fundo em escala de cinza e mover a foto recortada para o centro. Que tal tentar esta modificação?

AULA 5 - Brincando com *Chroma key*

Nesta aula, os estudantes criarão o efeito Chroma key em uma imagem, sobrepondo uma imagem sobre outra imagem de fundo e anulando uma cor padrão da imagem original, por exemplo, verde.

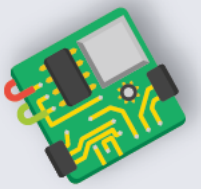
Objetivos	Aplicar operações matemáticas para realizar efeitos de imagem. Utilizar condicionais em efeitos de imagens. Criar funções com parâmetros para definir operações sobre imagens. Manipular imagens em Python.
Conteúdo	Efeitos de imagem: Chroma key. Propriedades dos pixels. Funções do JES: getPixels, get/setRed, get/setGreen, get/setBlue, getHeight, getWidth, get/setPixel, distance. Programação: variáveis, expressões aritméticas, condicionais, loops, listas e funções.
Atividades	Criando efeitos de Matiz.
Resultados de Aprendizagem	PC2. Usar passos básicos na solução de problemas algorítmicos para projetar soluções (por exemplo, declaração e exploração de problemas, exemplos de instâncias, design, implementação de uma solução, teste, avaliação). PC5. Avaliar maneiras pelas quais diferentes algoritmos podem ser usados para resolver o mesmo problema. PC8. Representar dados de várias formas, incluindo texto, sons, imagens e números. PC15. Examinar as conexões entre os elementos da matemática e da ciência da computação, incluindo números binários, lógica, conjuntos e funções. PCP10. Demonstrar compreensão dos algoritmos e sua aplicação prática. PCP11. Implementar soluções de problemas usando uma linguagem de programação, incluindo: comportamento de looping, instruções condicionais, lógica, expressões, variáveis e funções.
Materiais	JES. Folha de Atividades.

DESCRIÇÃO DAS ATIVIDADES

Nesta aula, os estudantes irão aplicar os conhecimentos construídos nas aulas sobre condicionais para criar efeito de *Chroma key*. O professor deve resgatar a aula de *Chroma key*, do ano anterior, e buscar, junto com os estudantes, relembrar a lógica deste efeito. O conceito do *Chroma key* é similar ao do recorte, pois copia uma cor de uma imagem para outra. Diferentemente do ano anterior, nesta aula, os estudantes utilizarão a distância entre cores RGB.

Para calcular a distância entre cores RGB, os estudantes devem utilizar a função *distance* do JES que calcula a distância euclidiana entre duas cores, onde cada componente de cor *red*, *green* ou *blue* é visto como uma coordenada, como se fossem os valores de *x*, *y* e *z* no espaço. O professor deve, inicialmente, demonstrar o uso do comando *distance* e, em seguida, pedir que os estudantes que façam a Folha de Atividades.

O código a seguir ilustra o efeito de *Chroma Key*. Esta função recebe três parâmetros: *imagem*, que é a imagem que será sobreposta, *fundo*, que é a imagem de fundo que substituirá o fundo padrão da primeira imagem, e *cor*, que é a cor padrão que será anulada da primeira imagem.

```
def chromakey(imagem, fundo, cor1):
  for x in range(1, getWidth(imagem)):
    for y in range(1, getHeight(imagem)):
      pixel = getPixel(imagem, x,y)
      if (distance( getColor(pixel), cor1 ) < 180):
        pixel2 = getPixel(fundo,x,y)
        setColor(pixel, getColor(pixel2) )
```

FOLHA DE ATIVIDADES

Implementando o efeito *Chroma key*

Vamos exercitar suas habilidades matemáticas e lógicas para a criação do efeito *Chroma key*.

- ✓ **Primeira parte.** Remova o fundo da imagem que você trouxe para a atividade. Lembre-se de ajustar o valor da cor (verde ou azul) para eliminar todo o fundo.

- ✓ **Segunda parte**. Agora que você já sabe remover uma cor padrão do fundo de uma imagem, experimente colar os pixels de outra imagem para compor o fundo, criando o efeito *Chroma key*. Você pode pensar em algum lugar especial para você.

- ✓ **Terceira parte:** Agora que você já está dominando a arte do *Chroma key*, que tal compor uma imagem com três imagens diferentes: *imagem 1 com fundo azul/verde, imagem 2 com fundo azul/verde e um fundo.*

- ✓ **Desafio**. É possível utilizar um ou mais efeitos combinados com o *Chroma key*? Exercite sua criatividade e crie um novo efeito.

AULA 6 - Gerador de Memes

Nesta aula, os estudantes utilizarão todos os conceitos aprendidos na atividade gerador de memes.

Objetivos	Aplicar operações matemáticas para realizar efeitos de imagem. Utilizar condicionais em efeitos de imagens. Utilizar funções com parâmetros em efeitos de imagens. Manipular imagens em Python.
Conteúdo	Efeitos de imagem. Funções do JES: getPixels, get/setRed, get/setGreen, get/setBlue, getHeight, getWidth, get/setPixel, distance. Programação: variáveis, expressões aritméticas, condicionais, loops, listas e funções.
Atividades	Gerador de Memes.
Resultados de Aprendizagem	PC2. Usar passos básicos na solução de problemas algorítmicos para projetar soluções (por exemplo, declaração e exploração de problemas, exemplos de instâncias, design, implementação de uma solução, teste, avaliação). PC5. Avaliar maneiras pelas quais diferentes algoritmos podem ser usados para resolver o mesmo problema. PC8. Representar dados de várias formas, incluindo texto, sons, imagens e números. PC15. Examinar as conexões entre os elementos da matemática e da ciência da computação, incluindo números binários, lógica, conjuntos e funções. PCP10. Demonstrar compreensão dos algoritmos e sua aplicação prática. PCP11. Implementar soluções de problemas usando uma linguagem de programação, incluindo: comportamento de looping, instruções condicionais, lógica, expressões, variáveis e funções.
Materiais	JES. Folha de Atividades.

DESCRIÇÃO DAS ATIVIDADES

Nesta aula, os estudantes irão utilizar todos os conceitos aprendidos durante a unidade para a criação do projeto final. O projeto final consiste em criar uma função, denominada geradorDeMemes, que criará memes divertidos com base nos parâmetros passados pelo programador. O gerador de memes deve aplicar efeitos como negativo, escala de cinza, matiz, bordas, adicionar nomes, *Chroma key* ou outros efeitos vistos durante esta unidade.

Ao término da aula, o professor deve selecionar três estudantes com os memes mais divertidos e complexos para apresentar o projeto para os colegas. É ideal que o professor comunique este momento no início da aula, gerando motivação entre os estudantes.

FOLHA DE ATIVIDADES - Gerador de Memes

1 – Escreva a sua proposta para o gerador de Memes.

Ex.: Vou adicionar o nome "MIGA, SUA LOUCA?!" na imagem e deixá-la em escala de cinza.

____
____
____
____
____
____
____
____
____

2 - Marque com um "X" os conceitos que utilizará:

[ ] loop sobre uma lista de pixels
[ ] loop dentro de loop
[ ] condicionais
[ ] funções
[ ] parâmetros em funções
[ ] operações aritméticas
[ ] operadores relacionais
[ ] outros: ____

3 - Marque com um "X" os efeitos que utilizará:

[ ] adicionar nomes
[ ] adicionar borda
[ ] matiz
[ ] negativo
[ ] escala de cinza
[ ] binário
[ ] recorte
[ ] *Chroma key*
[ ] outros: ____

4 – Crie o código para seu gerador de memes utilizando todos os conceitos e efeitos marcados.

AULA 2 – AULA 7 – Cidade na Lama

Nesta aula, os estudantes irão aprender como o conceito de grafo pode ser utilizado em atividades do dia a dia através de uma atividade desplugada.

Objetivos	Compreender o conceito de grafo. Utilizar estratégias de pensamento computacional para solucionar um problema de minimização de caminhos.
Conteúdo	Grafos não dirigidos. Algoritmos sobre grafos.
Atividades	Cidade na Lama.
Resultados de Aprendizagem	PC1. Descrever como uma simulação pode ser usada para resolver um problema. PC5. Avaliar maneiras pelas quais diferentes algoritmos podem ser usados para resolver o mesmo problema. PC6. Trabalhar com algoritmos de busca e classificação. PC9. Usar representações visuais de estados de problemas, estruturas e dados (por exemplo, gráficos, diagramas, diagramas de rede, fluxogramas).
Materiais	Folha de atividades.

DESCRIÇÃO DAS ATIVIDADES

Nesta aula, os estudantes irão realizar uma atividade desplugada relacionada a grafos. No início da aula, o professor deve apresentar o problema presente na Folha de Atividades. Deve contextualizar a atividade sem necessariamente fazer uma introdução sobre o conteúdo. O objetivo da atividade é fazer com que os alunos consigam encontrar caminhos mínimos entre vértices, representados na atividade como casas, de modo que seja possível visitar todas as casas através das arestas, minimizando o caminho percorrido.

Os estudantes devem elaborar suas próprias estratégias para alcançar o objetivo. O ideal é que eles verifiquem sempre se algum colega conseguiu concluir a atividade percorrendo um caminho menor, gerando uma competição saudável.

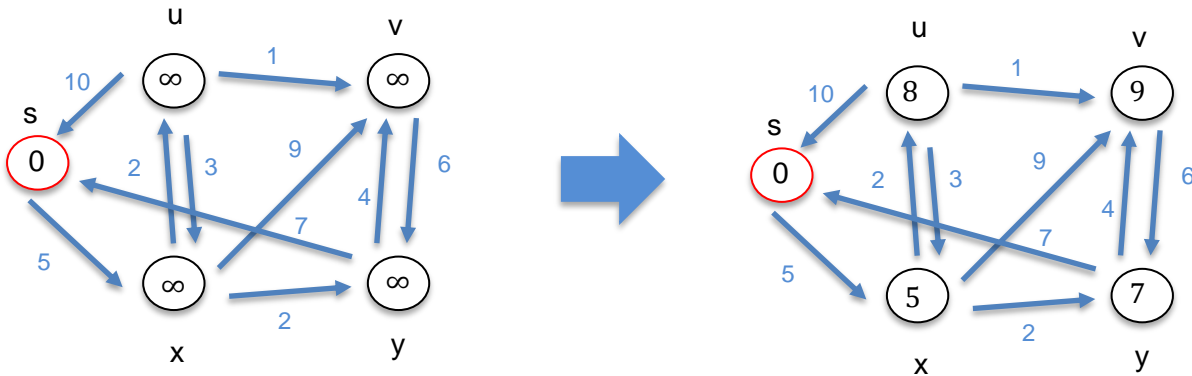
O professor pode dar dicas para os alunos como, por exemplo:

- Inicie com o mapa vazio, preencha cada caminho até uma casa que não foi visitada;
- Ou inicie com o mapa cheio e remova os caminhos desnecessários.

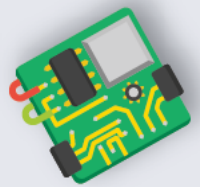
Ao término da atividade, o professor pode eleger alguns alunos que tenham tido boas estratégias para apresenta-las aos colegas.

Em seguida, o professor pode apresentar uma estratégia mais eficiente, representando a cidade na forma de um grafo e usando um algoritmo de caminho mais curto, como Dijkstra.

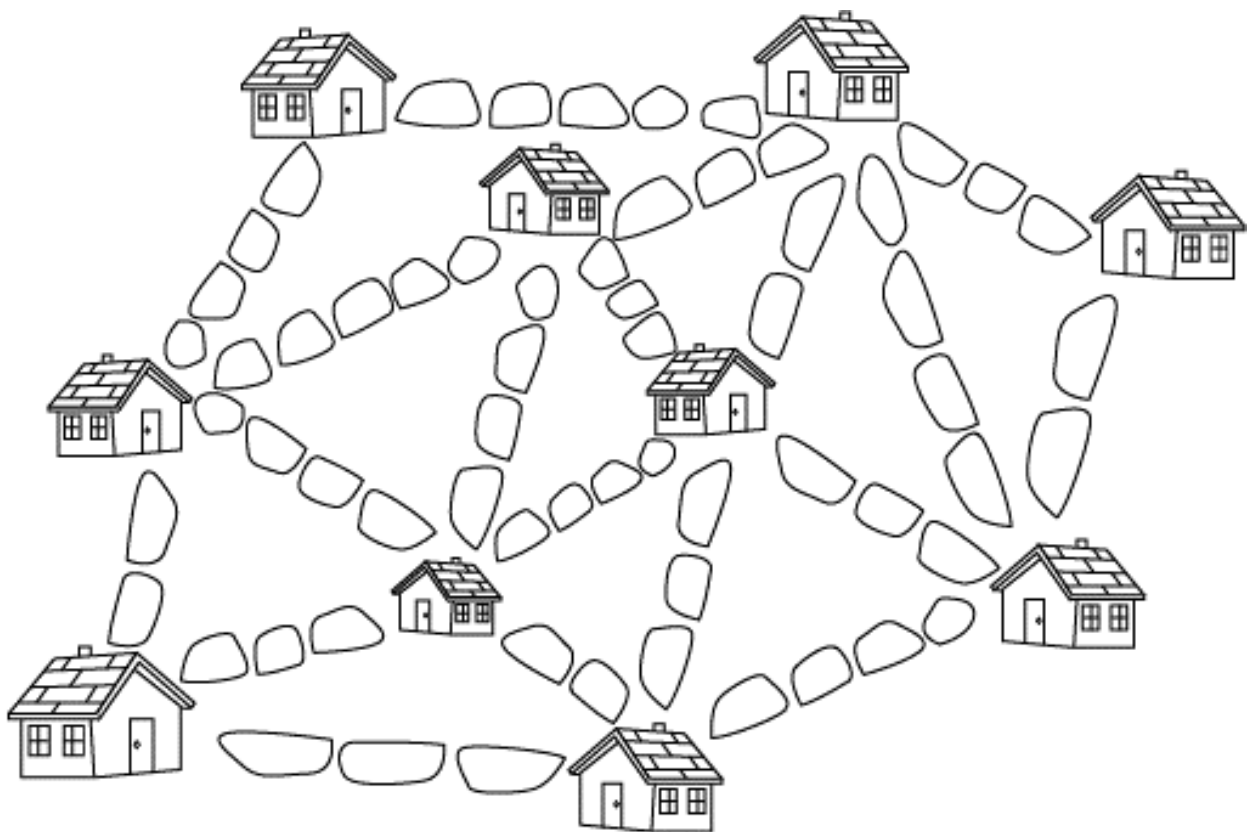
```
função Dijkstra(Grafo, origem):

    cria um conjunto de vértices Q
    para cada vértice v no Grafo:
        distancia[v] ← INFINITO
        precedente[v] ← INDEFINIDO
        adicionar v à Q

    distancia[origem] ← 0

    enquanto Q não estiver vazio:
        u ← vértice em Q com min dist[u]

        remove u de Q
        para cada v adjacente à u:
            if distancia[u] + peso(u, v) < distancia[v]:
                distancia[v] ← distancia[u] + peso(u, v)
                precedente [v] ← u

    retorna distancia[], precedente[]
```

Vejamos como ficaria o grafo inicial e final, com as distâncias calculadas.:

Esta tabela demonstra os pesos encontrados com o algoritmo de Dijkstra, partindo o vértice s.

Vértices	**s**	**u**	**v**	**x**	**y**
Distância	0	8	9	5	7
Precedente	s	x	u, x	s	x

Como conclusão, o professor pode levantar uma discussão sobre em quais situações, no dia a dia, poderíamos aplicar este tipo de estratégia: rede de energia, rede de computadores, distribuição de água, percurso de mapas e outros.

FOLHA DE ATIVIDADES – Cidade na Lama

Uma pequena cidade do Brasil sofre todos os anos com as fortes chuvas. Nesta cidade, sempre que chove, as pessoas têm dificuldade de sair de casa pois as ruas ficam cheias de lama. Por conta deste problema, o prefeito da cidade resolveu pavimentar as ruas. Mas, contando com poucos recursos, pediu para que os funcionários encontrassem uma solução mais barata para pavimentar as ruas. A solução deve permitir que todos os moradores possam sair de suas casas para qualquer outra casa através das ruas pavimentadas.

Esta imagem representa a cidade. O número de blocos de pavimentação entre cada casa representa o custo de pavimentação. Ou seja, se há dois blocos pavimentados entre as casas A e B, o custo desta pavimentação é 2. **Encontre o melhor percurso que ligue todas as casas, mas utilize a menor quantidade possível de blocos.**

Qual estratégia você usou para resolver o problema?

____

____

____

____

UNIDADE III

Python

AULA 1 – Fluxograma

Nesta aula, os estudantes aprenderão sobre fluxogramas através de atividades de organização e criação de fluxogramas.

Objetivos	Compreender o que são fluxogramas. Reconhecer os principais símbolos de um fluxograma. Aplicar o conceito de fluxograma para criação de algoritmos. Utilizar variáveis, operações aritméticas e lógicas, loops, estruturas condicionais e comandos de entrada e saída em fluxograma.
Conteúdo	Fluxogramas: • Terminal; • Seta de Fluxo; • Processamento; • Entrada de Dados; • Saída de Dados; • Decisão; • Conector; • Loops.
Atividades	Fluxograma.
Resultados de Aprendizagem	PC2. Usar passos básicos na solução de problemas algorítmicos para projetar soluções (por exemplo, declaração e exploração de problemas, exemplos de instâncias, design, implementação de uma solução, teste, avaliação). PC4. Definir um algoritmo como uma sequência de instruções que pode ser processada por um computador. PC8. Representar dados de várias formas, incluindo texto, sons, imagens e números. PC9. Usar representações visuais de estados de problemas, estruturas e dados (por exemplo, gráficos, diagramas, diagramas de rede, fluxogramas). PC16. Fornecer exemplos de aplicações interdisciplinares do pensamento computacional. PCP5. Reunir e manipular dados usando uma variedade de ferramentas digitais. PCP10. Demonstrar compreensão dos algoritmos e sua aplicação prática.
Materiais	Folha de Atividades. Papel. Caneta. Cola

DESCRIÇÃO DAS ATIVIDADES

Nesta aula, os estudantes reconhecerão os símbolos que compõem um Fluxograma e criarão algoritmos simples.

1ª Parte: O professor deve iniciar a aula relembrando o conceito de programas e algoritmos. Logo após, o professor pede para os alunos falarem quais foram as formas que eles já utilizaram para criar algoritmos. A partir daí, o professor discute sobre estas várias formas de representar um algoritmo: linguagem de blocos, linguagem textual e outros. Por fim, o professor inicia a apresentação de uma nova forma de representação: o fluxograma. O professor deve, na lousa, apresentar os principais símbolos: **Terminal, Seta de Fluxo, Processamento, Entrada e Saída de Dados, Decisão e Conector**.

Terminal: inicio / fim de um fluxograma

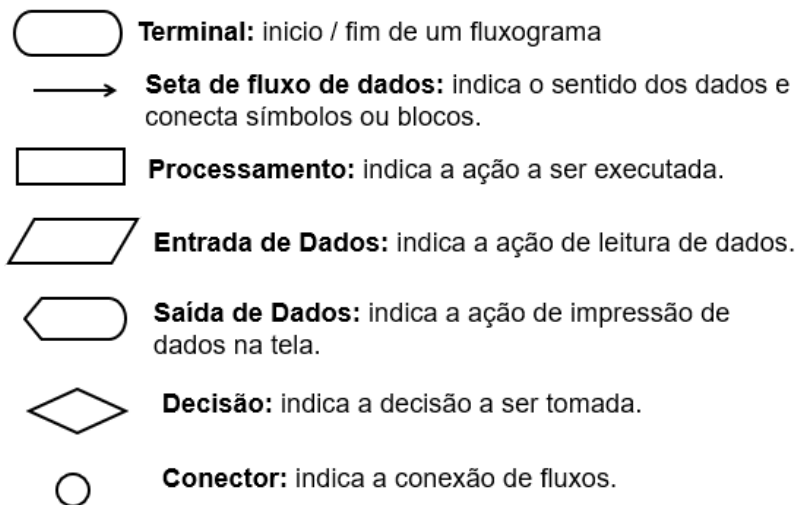
Seta de fluxo de dados: indica o sentido dos dados e conecta símbolos ou blocos.

Processamento: indica a ação a ser executada.

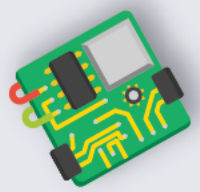
Entrada de Dados: indica a ação de leitura de dados.

Saída de Dados: indica a ação de impressão de dados na tela.

Decisão: indica a decisão a ser tomada.

Conector: indica a conexão de fluxos.

Em seguida, o professor apresenta um exemplo de fluxograma para expressar um algoritmo para calcular, por exemplo, a área de um retângulo. O algoritmo deve ler a base e altura do dispositivo de entrada, calcular a área e armazená-la numa variável e, em seguida, apresentar o valor da área no dispositivo de saída. Por fim, o professor deve alterar o fluxograma para checar adicionalmente se o retângulo é também um quadrado, usando uma estrutura de decisão.

2ª Parte: Em um segundo momento, o professor divide a turma em grupos de três estudantes e pede para que eles façam a primeira parte da Folha de Atividades. Os estudantes devem recortar os símbolos e organizá-los, segundo a descrição.

3ª Parte: Os estudantes, agora de forma individual, devem realizar a segunda parte da Folha de Atividades. O objetivo é que os estudantes criem os fluxogramas em papel.

TÓPICOS IMPORTANTES

O fluxograma é mais uma forma de representar os algoritmos. No fluxograma, existem vários símbolos que funcionam como comando no Scratch ou como palavras reservadas do Python. Com exceção da Decisão e do Conector, esses símbolos se conectam a penas um outro símbolo. A Seta indica a ordem de execução de cada bloco, iniciando sempre o Início em direção ao final. Loops são representados pela ligação de um bloco com outro anterior, através da Seta de Fluxo.

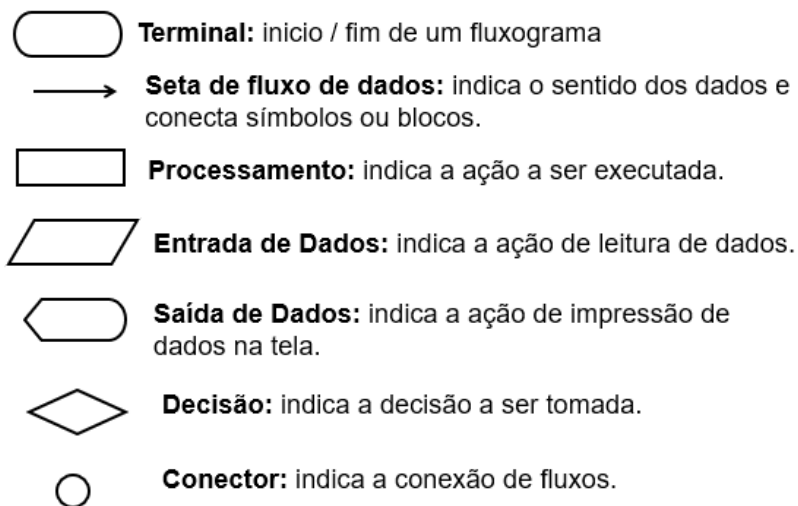
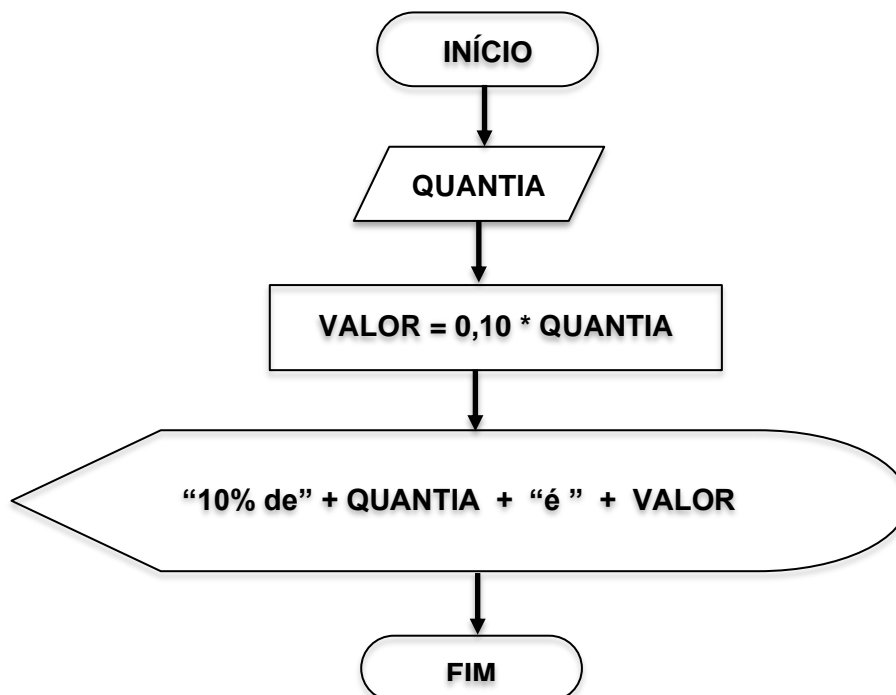



Figura 1 -Símbolos do Fluxograma

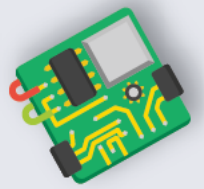
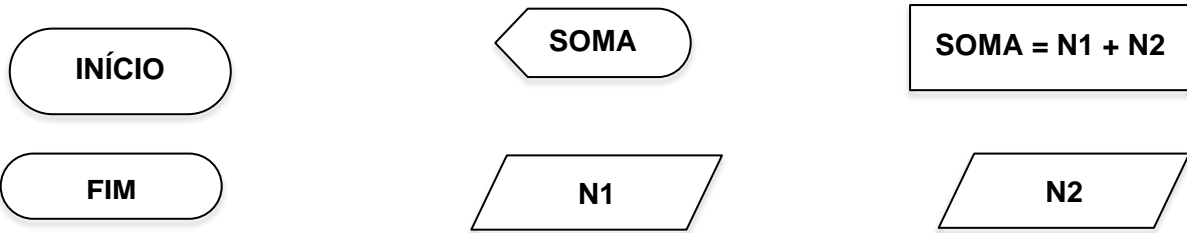
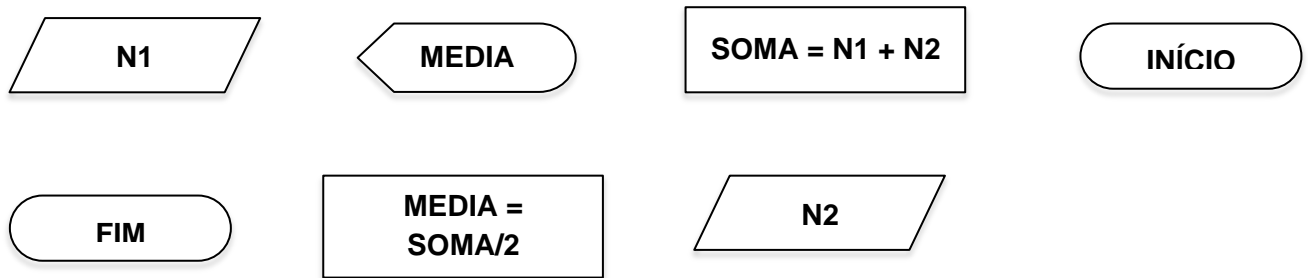
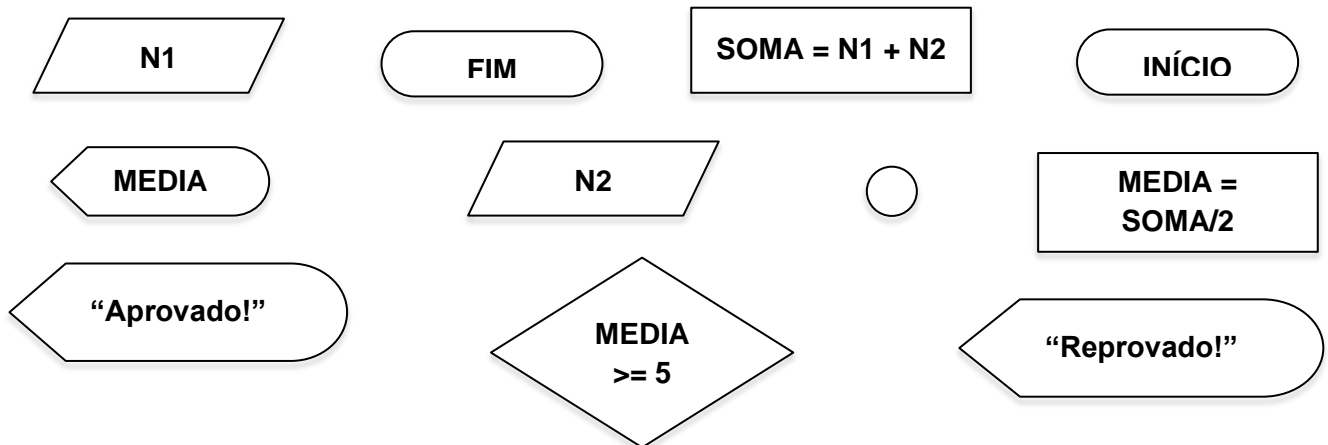
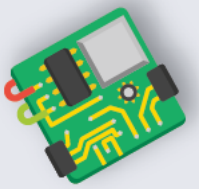
Vamos ver um exemplo que calcula e exibe 10% de uma quantia informada pelo usuário (N1). Caso o usuário informe o valor 100, a mensagem será “10% de 100 é 10”.

FOLHA DE ATIVIDADES – parte I

Agora que você sabe tudo sobre fluxogramas, exercite um pouco com esta atividade. Utilize os símbolos disponíveis na folha mais à frente. Recorte-os, cole-os e desenhe setas para completar os fluxogramas, indicando o fluxo de execução.

a) Leia dois números, some-os e exiba o resultado.

b) Leia dois números, calcule e exiba a média deles.

c) Leia duas notas, calcule a média delas e verifique se a média é maior ou igual 5 (aprovação) ou menor que 5 (reprovação).

d) Solicite um nome e exiba-o após a frase “Seu nome é “. O fluxograma deve continuar solicitando e mostrando os nomes até que o nome digitado seja a palavra “Sair”.

Recorte os símbolos e os organize no espaço reservado.

FOLHA DE ATIVIDADES – parte II

Individualmente, desenhe os seguintes fluxogramas ou utilize um programa de computador.

a) Crie um algoritmo que calcule a área de um quadrado, dado o lado do quadrado.

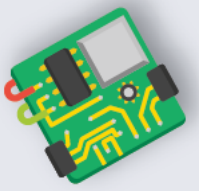
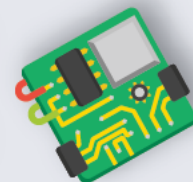
b) Crie um algoritmo que calcule a área de um triângulo, dada a base e a altura.

c) Crie um algoritmo que receba os valores dos três ângulos internos de um triângulo e diga se ele é equilátero.

d) Crie um algoritmo que solicite o nome de uma figura. Em seguida, exiba este nome cem vezes na tela.

AULA 2 – Apertadinho

Nesta aula, os estudantes farão o jogo Apertadinho utilizando conceitos como variáveis, entrada e saída, condicionais e loops.

Objetivos	Desenvolver programas em Python que trabalhem com entrada e saída de dados textuais.
	Aplicar o conceito de loops para automatizar tarefas repetitivas.
	Utilizar estruturas condicionais para realizar escolhas do usuário e modificar o fluxo do programa de acordo com estas escolhas.
Conteúdo	Variáveis.
	Atribuição.
	Entrada e saída.
	Loops.
	Condicionais.
	Expressões lógicas e relacionais.
Atividades	Jogo Apertadinho
Resultados de Aprendizagem	PC2. Usar passos básicos na solução de problemas algorítmicos para projetar soluções (por exemplo, declaração e exploração de problemas, exemplos de instâncias, design, implementação de uma solução, teste, avaliação).
	PC4. Definir um algoritmo como uma sequência de instruções que pode ser processada por um computador.
	PC8. Representar dados de várias formas, incluindo texto, sons, imagens e números.
	PC9. Usar representações visuais de estados de problemas, estruturas e dados (por exemplo, gráficos, diagramas, diagramas de rede, fluxogramas).
	PC15. Examinar as conexões entre os elementos da matemática e da ciência da computação, incluindo números binários, lógica, conjuntos e funções.
	PC16. Fornecer exemplos de aplicações interdisciplinares do pensamento computacional.
	PCP5. Reunir e manipular dados usando uma variedade de ferramentas digitais.
	PCP10. Demonstrar compreensão dos algoritmos e sua aplicação prática.
	PCP11. Implementar soluções de problemas usando uma linguagem de programação, incluindo: comportamento de looping, instruções condicionais, lógica, expressões, variáveis e funções.

Materiais	JES instalado. projeto1_unidade3_aula2.py. Folha de Atividades.

DESCRIÇÃO DAS ATIVIDADES

Nesta aula, os estudantes farão o jogo Apertadinho, utilizando conceitos de variáveis, entrada e saída, condicionais, loops e expressões condicionais e lógicas.

1ª Parte: O professor deve iniciar a aula perguntando aos estudantes se eles conhecem o jogo Apertadinho, às vezes chamado por outro nome como "Adivinhe o número". Caso alguns alunos conheçam, o professor deve pedir para que eles expliquem de modo a compartilhar experiências e verificar se é a compreensão correta do jogo. Logo após, o professor deve simular uma rodada do jogo Apertadinho com os alunos, utilizando a lousa para marcar os números máximo e mínimo. O professor deve verbalizar uma simulação dos passos a seguir, descrevendo a execução do algoritmo como o usuário a perceberia.

1. O professor sorteia um número entre os valores mínimo e máximo (por exemplo, entre 0 e 100). Alternativamente, ele pode pedir a um dos alunos para escolher este número e ser o responsável pela verificação.
2. Um dos alunos dá um palpite entre os valores mínimo e máximo.
3. Verificar se é o número sorteado.
    a. Caso positivo, o jogador ganhou!
    b. Caso negativo, verifica-se se:
        i. O palpite é maior que o mínimo e menor que o número sorteado.
            1. Caso positivo, o palpite passa a ser o novo mínimo!
        ii. O palpite é menor que o máximo e maior que o número sorteado.
            1. Caso positivo, o palpite passa a ser o novo máximo!
        iii. O máximo e mínimo são separados por apenas um número.
            1. Caso positivo, o jogador perdeu!
4. Os passos anteriores são repetidos até que o número seja encontrado ou os números máximo e mínimo estejam separados por apenas um número (ex: 53 e 55).

2ª Parte: O professor então deve solicitar que os alunos que se organizem em duplas ou trios e formulem a lógica do jogo por ele simulado. Desta forma, os alunos devem construir o fluxograma do jogo. Essa etapa deve durar até 30 minutos de modo a não prejudicar a etapa posterior.

3ª Parte: Os estudantes devem executar a Folha de Atividades criando o jogo Apertadinho na linguagem Python, com base na explicação do professor e a partir do fluxograma criado.

O professor deve circular entre os grupos para tirar dúvidas quando solicitado ou quando perceber dificuldades excessivas entre os estudantes.

FOLHA DE ATIVIDADES

O jogo Apertadinho consiste em ter um número sorteado entre um número mínimo e um número máximo (por exemplo, entre 0 e 100). A cada rodada, o jogador dá um palpite no intuito de acertar qual foi o número sorteado. A cada rodada, também, é reduzido o limite dos números mínimo e máximo para o jogador, "apertando" cada vez mais as possibilidades. Os critérios para esta redução são:

a) Se o palpite é maior que o mínimo e menor que o número sorteado, ele será o novo mínimo.
b) Se o palpite é menor que o máximo e maior que o número sorteado, ele será o novo máximo.

Vamos a um exemplo: O número sorteado é 50, o mínimo é 0 e o máximo é 100. O jogador escolhe o valor 40. Assim, o 40 será o novo mínimo e o máximo continua sendo 100. Na próxima rodada, o jogador escolhe o valor 60. Agora o novo máximo passará a ser 60. Deste modo, o jogador só poderá escolher números entre 40 e 60 na próxima jogada.

No jogo, o jogador ganha se acertar o número, mas perde se os números mínimo e máximo forem separados por apenas um número, ou seja, separados apenas pelo número sorteado.

Vamos a mais um exemplo: Os limites são 40 e 60. O jogador escolhe o número 51! Como 51 é maior que 50 e menor que 60 (máximo), os novos limites são 40 e 51. Na rodada seguinte o jogador escolhe o número 49. Logo, o limite mínimo será agora 49 e o máximo permanece 51. A única alternativa possível é o número 50, logo o jogador não conseguiu descobrir o valor.

Agora, vamos lá, com base em seus conhecimentos de programação e nas dicas dadas sobre o jogo Apertadinho:

1 - Crie um fluxograma que represente as regras do jogo apertadinho.

2 - Crie um programa que implemente o jogo apertadinho utilizando a linguagem Python.

AULA 3 – Enquetes

Nesta aula, os estudantes farão um programa de enquetes, reforçando a aprendizagem dos conceitos de variáveis, entrada e saída, condicionais, loops, e adicionando o conceito de contadores.

Objetivos	Desenvolver programas em Python que trabalhem com entrada e saída de dados textuais.
	Aplicar o conceito de loops para automatizar tarefas repetitivas.
	Utilizar estruturas condicionais para realizar escolhas do usuário e modificar o fluxo do programa de acordo com estas escolhas.
	Implementar expressões aritméticas simples para o cálculo de percentuais.
Conteúdo	Variáveis.
	Entrada e saída.
	Loops.
	Condicionais.
	Contadores.
	Expressões aritméticas.
Atividades	Sistema de enquetes: Quem é mais popular?
Resultados de Aprendizagem	PC2. Usar passos básicos na solução de problemas algorítmicos para projetar soluções (por exemplo, declaração e exploração de problemas, exemplos de instâncias, design, implementação de uma solução, teste, avaliação).
	PC4. Definir um algoritmo como uma sequência de instruções que pode ser processada por um computador.
	PC8. Representar dados de várias formas, incluindo texto, sons, imagens e números.
	PC15. Examinar as conexões entre os elementos da matemática e da ciência da computação, incluindo números binários, lógica, conjuntos e funções.
	PC16. Fornecer exemplos de aplicações interdisciplinares do pensamento computacional.
	PCP5. Reunir e manipular dados usando uma variedade de ferramentas digitais.
	PCP10. Demonstrar compreensão dos algoritmos e sua aplicação prática.
	PCP11. Implementar soluções de problemas usando uma linguagem de programação, incluindo: comportamento de looping, instruções condicionais, lógica, expressões, variáveis e funções.

	PCP15. Coletar e analisar dados que são produzidos a partir de várias execuções de um programa de computador.
Materiais	JES instalado. projeto1_unidade3_aula3.py. Folha de Atividades.

DESCRIÇÃO DAS ATIVIDADES

Nesta aula, os estudantes farão um programa de enquetes, reforçando a aprendizagem dos conceitos de variáveis, entrada e saída, condicionais e loops, e adicionado os novos conceitos de contadores e expressões aritméticas.

1ª Parte: O professor deve iniciar a aula perguntando que tipos de comandos foram trabalhados na linguagem Python até então. Os estudantes devem voluntariamente compor uma lista com entrada, saída, loops, condicionais. É importante que, em meio à reflexão, os estudantes descrevam para que estes comandos podem ser utilizados e como isto foi feito nas aulas anteriores.

2ª Parte: O professor deve falar que o tema de hoje é uma espécie de desafio, onde os estudantes criarão um sistema de enquetes. O professor pode executar o programa projeto1_unidade3_aula3, que consiste em uma enquete para saber que meio de transporte é mais utilizado pelos estudantes. Ao executar o programa, solicitando que os estudantes respondam a enquete e apresentado os resultados, o professor deve perguntar que tipos de comandos são necessários para criar aquele programa. O professor deve abrir um novo arquivo no editor e, junto com os estudantes, criar um programa similar de enquetes. É importante que os estudantes participem da construção do programa, oferecendo dicas voluntárias de implementação. O professor pode organizar a construção do programa em etapas, resolvendo os subproblemas de:

- inicializar o contador;
- fazer o loop para obter os dados;
- contar os dados;
- calcular os percentuais;
- apresentar os resultados.

3ª Parte: Os estudantes devem criar um programa de enquetes, seguindo as sugestões da Folha de Tarefas. O professor deve reservar um tempo da aula para implementação e teste dos programas. Se possível, ao final da aula, os estudantes podem apresentar seus programas.

FOLHA DE ATIVIDADES

Uma enquete é uma pesquisa de opinião, ou seja, um levantamento estatístico da opinião pública a respeito de algum tema. Muitas pesquisas de opinião trabalham com dados quantitativos e executam estatísticas sobre estes dados. Um exemplo é uma enquete com o objetivo de sondar os estudantes sobre qual meio de transporte eles utilizam para chegar à escola. Os participantes que respondem a essa enquete geram uma contagem para cada meio de transporte. Cálculos como a porcentagem de uso de cada meio de transporte podem ser realizados sobre estes dados, oferecendo uma visão ampla sobre a realidade social dos estudantes.

Exercício 1 - Quem é mais popular?

Vamos descobrir quem ou o que é mais popular na sua turma. Selecione abaixo um tipo de enquete:

( ) Qual sua série de tv favorita?
( ) Qual seu filme favorito?
( ) Qual seu gênero de filmes favorito?
( ) Qual seu gênero musical favorito?
( ) Qual seu(sua) cantor(a) favorito(a)?
( ) Qual sua banda favorita?
( ) Qual seu(sua) youtuber favorito(a)?

Descreva as opções da sua enquete:

Opção 1: ____

Opção 2: ____

Opção 3: ____

Opção 4: ____

Opção 5: ____

Opção 6: Nenhuma das opções anteriores.

Crie um programa que implemente sua enquete utilizando a linguagem Python.

AULA 4 – Criptografia de César

Nesta aula, os estudantes farão um programa que implementa o método de criptografia chamado cifra de César.

Objetivos	Desenvolver programas em Python que trabalhem com entrada e saída de dados textuais.
	Aplicar o conceito de loops para automatizar tarefas repetitivas.
	Utilizar estruturas condicionais para tomar decisões a partir do valor dos dados.
	Manipular strings em Python através de índices e de loops.
Conteúdo	Variáveis.
	Entrada e saída.
	Loops.
	Condicionais.
	Strings.
	Indexação de strings.
	Criptografia.
Atividades	Criptografia de César.
Resultados de Aprendizagem	PC2. Usar passos básicos na solução de problemas algorítmicos para projetar soluções (por exemplo, declaração e exploração de problemas, exemplos de instâncias, design, implementação de uma solução, teste, avaliação).
	PC4. Definir um algoritmo como uma sequência de instruções que pode ser processada por um computador.
	PC8. Representar dados de várias formas, incluindo texto, sons, imagens e números.
	PC15. Examinar as conexões entre os elementos da matemática e da ciência da computação, incluindo números binários, lógica, conjuntos e funções.
	PC16. Fornecer exemplos de aplicações interdisciplinares do pensamento computacional.
	PCP5. Reunir e manipular dados usando uma variedade de ferramentas digitais.
	PCP10. Demonstrar compreensão dos algoritmos e sua aplicação prática.

	PCP11. Implementar soluções de problemas usando uma linguagem de programação, incluindo: comportamento de looping, instruções condicionais, lógica, expressões, variáveis e funções.
Materiais	JES instalado. projeto1_unidade3_aula4.py. Folha de Atividades.

DESCRIÇÃO DAS ATIVIDADES

Nesta aula, os estudantes farão um programa que implementa o método de criptografia chamado cifra de César.

1ª Parte: O professor deve iniciar a aula mostrando o programa do projeto1_unidade3_aula5. Este programa implementa a criptografia de César para converter um texto normal em um texto cifrado. O professor deve executar o programa, incluindo a cifragem e a decifragem. Ao final, o professor deve, junto com os estudantes, descrever como funciona a criptografia de César. Os estudantes já trabalharam com este tema em atividades de computação desplugada, disponível no livro do 7º ano.

2ª Parte: Os estudantes devem implementar o programa individualmente, seguindo os desafios previstos na Folha de Atividades. O professor deve oferecer ajuda individual, quando solicitado. É importante salientar que os estudantes irão apresentar dúvidas sobre como uma mensagem em formato de texto é tratada. Quando isso acontecer, convém explicar como as *strings* são tratadas como listas que iniciam na posição 0. O professor deve estimular os estudantes para a leitura dos Tópicos Relevantes sobre *strings*, necessário para o desenvolvimento do programa.

3ª Parte: Ao final da aula, os estudantes devem brincar, trocando mensagens criptografadas entre si.

TÓPICOS RELEVANTES

A linguagem de programação Python é uma das linguagens mais fáceis de aprender quando se é novato na programação de computadores. Mesmo assim, é preciso prestar atenção em alguns aspectos para que seus programas funcionem corretamente.

Quando criamos uma variável em um programa em Python, esta variável pode ser de vários tipos: número inteiro, número real, texto, dentre outros tipos. Lembre-se que a variável é uma caixinha que guarda um valor. Em Python, uma variável pode armazenar diferentes tipos de valores ao longo da execução do programa. Mas, se em um determinado momento a caixinha está guardando um número, você não pode executar, sobre este valor, funções que são específicas para textos, por exemplo. Apenas quando a variável guarda um texto, é possível usar funções específicas para textos.

Já trabalhamos com números, mas hoje vamos trabalhar com variáveis textuais, por isso, vamos estudar um pouco o que são *strings* e como manipulá-las:

Strings (str)

Uma ***string*** representa um texto, ou seja, um conjunto encadeado de caracteres. Caractere, por sua vez, é um símbolo gráfico. São exemplos de caracteres as letras do alfabeto, os algarismos, sinais de pontuação de texto, dentre outros. Em uma *string*, os caracteres estão dispostos em uma determinada ordem, e esta ordem importa bastante. Por exemplo, uma variável que guarda o nome 'Bianca' só faz sentido para o usuário respeitando a ordem das letras do nome, ou seja, o seu encadeamento.

Strings em Python são chamadas de **str**. Uma variável *string*, ou seja, uma variável do tipo str, representa um texto com zero ou mais caracteres, que devem ser representados entre apóstrofos ou aspas. Exemplos: 'a', 'casa', "carro", "z", 'fl', 'Bianca é linda', ' ', '', "", '+', '#$%@$%', 'qwerty1234!@#$'.

Operações com *strings* em Python

Vamos manipular algumas *strings* como exemplo. Para começar, vamos armazenar algumas frases em variáveis. Exemplo:

```
frase1 = 'Bianca é linda'
frase2 = 'Luís é inteligente'
frase3 = 'Roberto é legal'
```

Podemos somar estas variáveis. Somar *strings* significa concatená-las, ou seja, colocá-las uma após a outra. Por exemplo:

```
frase4 = frase1 + frase2 + frase3
```

O resultado guardado em `frase4` será 'Bianca é lindaLuís é inteligenteRoberto é legal', que não estará perfeito, pois a soma simplesmente junta as frases, sem considerar os espaços que seriam necessários para compreendermos melhor o texto. Podemos manipular esta soma colocando caracteres específicos entre as frases. Por exemplo:

```
frase4 = frase1 + '. ' + frase2 + '. ' + frase3 + '.'
```

O resultado será: 'Bianca é linda. Luís é inteligente. Roberto é legal.'
Podemos acessar elementos específicos de uma *string*, considerando que o primeiro caractere está na posição 0, o segundo caractere está na posição 1, o terceiro caractere está na posição 2, e assim por diante. Por exemplo:

```
frase = 'Bianca é linda'
letra1 = frase[0]
letra2 = frase[12]
```

O valor na variável `letra1` será 'B', que é o primeiro caractere da *string* `frase`, enquanto que o valor na variável `letra2` será 'd', que é o penúltimo caractere da *string* `frase`. Estas posições dentro de uma *string* são também chamadas de índices.

Podemos percorrer *strings* através de comandos do tipo `for`. Por exemplo:

```
alfabeto = 'ABCDEFGHIJKLMNOPQRSTUVWXYZ'

for letra in alfabeto:
    showInformation('O caractere atual é: '  + letra)
```

O laço `for` vai percorrer todos os caracteres do alfabeto e, em cada execução, vai guardar um dos caracteres da variável `alfabeto` na variável `letra`. O laço for acima pode ser interpretado como "para cada caractere na variável `alfabeto`, repita o comando `showInformation()`".

Este mesmo esquema com o laço `for` nos permite procurar por um determinado caractere em uma *string*. Por exemplo:

```
frase = 'a bola rola'

for letra in frase:
    if letra == 'a':
        showInformation('Encontrei a letra a no texto!' )
```

Assim, o laço `for` vai percorrer todos os caracteres da variável `frase` e, em cada vez, vai verificar se o caractere é igual à letra ‘a’. Se verdadeiro, vai mostrar a mensagem.

FOLHA DE ATIVIDADES

A **cifra de César** é uma das primeiras técnicas de criptografia de que se tem registro na história. Recebeu este nome em homenagem ao conquistador romano Júlio César, que a empregava para proteger importantes mensagens políticas ou militares. Nesta técnica, cada letra da mensagem é substituída por outra que aparece no alfabeto após a letra original, saltando um número fixo de vezes. Os desafios a seguir ajudarão você a implementar um programa, na linguagem Python, que faz o processo de cifragem e decifragem de acordo com a criptografia de César.

Desafio 1 - Solicitando informações

Você precisa solicitar informações do usuário, como:

- se o usuário deseja cifrar ou decifrar a mensagem que inseriu no programa,
- a chave da criptografia usada e
- a mensagem a ser cifrada ou decifrada.

Para solicitar as informações, utilize o comando `requestString`, como no exemplo abaixo:

```
texto = requestString('Digite a mensagem (apenas letras maiusculas):')
```

Desafio 2 – Definindo informações

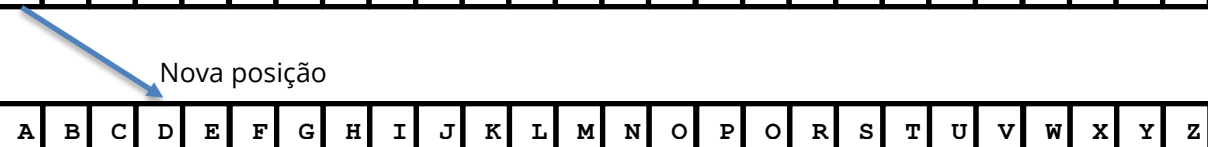
Você precisa definir uma *string* para ser o alfabeto e uma *string* vazia para armazenar a mensagem cifrada ou decifrada.

Desafio 3 – Implementando a Cifragem

O processo de cifragem da criptografia de César segue a lógica a seguir:

Chave: 3

Mensagem original: **EU AMO ESTUDAR**

Alfabeto normal

Mensagem cifrada: **HX DPR HVWXGDR**

Você pode implementar a cifragem através do algoritmo a seguir para cada caractere da mensagem inserida:

Passo 1: Se o caractere for um espaço ou qualquer outro que não seja uma letra do alfabeto, você deve adicioná-lo diretamente ao final da mensagem cifrada

```
texto_convertido = texto_convertido + caractere
```

Passo 2: Se o caractere não for um espaço:

Passo 2.1: Encontre o número dele no alfabeto normal.

```
num = alfabeto.find(caractere)
```

Passo 2.2: Adicione a chave de criptografia a este número e encontre a posição do caractere substituto.

```
posicao = num + chave
```

Passo 2.3: Verifique se o resultado da posição não ultrapassa a quantidade de letras no alfabeto. Se ultrapassar, você deve implementar uma lógica para encontrar a posição correta no alfabeto cifrado.

Passo 2.4: Adicione o caractere ao final da string que armazena a mensagem cifrada

```
texto_convertido = texto_convertido + alfabeto[posicao]
```

Desafio 4 – Implementando a Decifragem

O processo de decifragem da criptografia de César segue a lógica a seguir:

Mensagem cifrada: **WJH YDV**
Chave: 5
Alfabeto cifrado

Posição do alfabeto normal

Mensagem decifrada: **BOM DIA**

Você pode implementar a decifragem através do algoritmo a seguir para cada caractere da mensagem inserida:

Passo 1: Se o caractere for um espaço ou qualquer outro que não seja uma letra do alfabeto, você deve adicioná-lo diretamente ao final da mensagem cifrada

```
texto_convertido = texto_convertido + caractere
```

Passo 2: Se o caractere não for um espaço:

Passo 2.1: Encontre o número dele no alfabeto normal.

```
num = alfabeto.find(caractere)
```

Passo 2.2: Subtraia a chave de criptografia a este número e encontre a posição do caractere substituto.

```
posicao = num - chave
```

Passo 2.3: Verifique se o resultado da posição é menor que zero. Se for, você deve implementar uma lógica para encontrar a posição correta no alfabeto.

Passo 2.4: Adicione o caractere ao final da string que armazena a mensagem cifrada

```
texto_convertido = texto_convertido + alfabeto[posicao]
```

AULA 5 – Jogo da Forca

Nesta aula, os estudantes desenvolverão o jogo da Forca utilizando conceitos de strings, indexação e contadores.

Objetivos	Desenvolver programas em Python que trabalhem com entrada e saída de dados textuais.
	Aplicar o conceito de loops para automatizar tarefas repetitivas.
	Utilizar estruturas condicionais para verificação de igualdade entre caracteres.
	Implementar operações lógicas para verificação de mecanismos do jogo.
	Utilizar indexação de strings para identificação de caracteres separados de um texto.
Conteúdo	Variáveis.
	Entrada e saída.
	Loops.
	Condicionais.
	Expressões lógicas.
	Strings.
	Indexação de strings.
Atividades	Jogo da Forca
Resultados de Aprendizagem	PC2. Usar passos básicos na solução de problemas algorítmicos para projetar soluções (por exemplo, declaração e exploração de problemas, exemplos de instâncias, design, implementação de uma solução, teste, avaliação).
	PC4. Definir um algoritmo como uma sequência de instruções que pode ser processada por um computador.
	PC8. Representar dados de várias formas, incluindo texto, sons, imagens e números.
	PC15. Examinar as conexões entre os elementos da matemática e da ciência da computação, incluindo números binários, lógica, conjuntos e funções.
	PC16. Fornecer exemplos de aplicações interdisciplinares do pensamento computacional.
	PCP5. Reunir e manipular dados usando uma variedade de ferramentas digitais.
	PCP10. Demonstrar compreensão dos algoritmos e sua aplicação prática.
	PCP11. Implementar soluções de problemas usando uma linguagem de programação, incluindo: comportamento de looping, instruções condicionais, lógica, expressões, variáveis e funções.

	PCP15. Coletar e analisar dados que são produzidos a partir de várias execuções de um programa de computador.
Materiais	JES instalado. projeto1_unidade3_aula5.py. Folha de Atividades.

DESCRIÇÃO DAS ATIVIDADES

Nesta aula, os estudantes farão o jogo da Forca, o conhecido jogo de adivinhação de palavras, utilizando conceitos já conhecidos como variáveis, entrada e saída, condicionais, laços e expressões lógicas, além da utilização de indexação de *strings*.

1ª Parte: O professor deve iniciar a aula colocando pequenos traços na lousa, representando letras de uma palavra escolhida pelo professor. O professor então deve indagar a turma sobre o que representam aqueles traços. A ideia é que os alunos identifiquem que é o jogo da Forca. O professor pode explicar o funcionamento do jogo ou pedir que algum aluno que saiba, explique. O professor pode ainda simular o jogo da Forca com os alunos para exemplificar.

2ª Parte: Os estudantes devem, através da Folha de Atividades, implementar o jogo da Forca através de um programa em Python utilizando principalmente os conceitos de laços, condicionais e *strings*. O professor deve estimular os estudantes para a leitura dos Tópicos Relevantes sobre *strings*, necessário para o desenvolvimento do programa.

TÓPICOS RELEVANTES

Da aula anterior, já sabemos bastante sobre ***strings***. Vale lembrar que podemos acessar elementos específicos de uma *string*, considerando que o primeiro caractere está na posição 0, o segundo caractere está na posição 1, o terceiro caractere está na posição 2, e assim por diante. Por exemplo:

```
frase = 'Bianca é linda'
letra1 = frase[0]
letra2 = frase[12]
```

O valor na variável `letra1` será 'B', que é o primeiro caractere da *string* `frase`, enquanto que o valor na variável `letra2` será 'd', que é o penúltimo caractere da *string* `frase`. Estas posições dentro de uma *string* são também chamadas de índices.

Aprendemos na aula anterior que o laço `for` nos permite procurar por um determinado caractere em uma *string*. Por exemplo:

```
frase = 'a bola rola'

for letra in frase:
    if letra == 'a':
        showInformation('Encontrei a letra a no texto!' )
```

Assim, o laço `for` vai percorrer todos os caracteres da variável `frase` e, em cada vez, vai verificar se o caractere é igual à letra 'a'. Se verdadeiro, vai mostrar a mensagem.

Percorrendo *strings* através de índices

Além deste modo de percorrer uma *string* usando for, podemos utilizar o valor do índice da *string*. Este índice assume os valores de 0 até o tamanho da *string* subtraído de um. Por exemplo, a *string* 'Bola' tem tamanho 4 pois tem 4 caracteres. É possível variar os índices desta *string* de 0 até 3 [0 é 'B', 1 é 'o', 2 é 'l', 3 é 'a'].

A função `len()`, que retorna o tamanho da *string*, pode ajudar neste processo.
Por exemplo, se `nome = 'Ada'` e `sobrenome = 'Lovelace'`, a chamada à função `len(nome)` deve retornar o valor 4, enquanto que `len(sobrenome)` deve retornar o valor 10.

A função `range()`, que fornece uma lista de números de 0 até o valor passado para a função subtraído de um, também pode ser útil neste processo.
Por exemplo, a chamada à função `range(5)` retorna a lista de cinco números de 0 a 4, ou seja: `[0, 1, 2, 3, 4]`.

Combinando `range()` com `len()`, podemos saber os índices dos caracteres que precisamos percorrer em uma string. Assim, se `sobrenome = 'Lovelace'`, a chamada a `range(len(sobrenome))` deve gerar uma lista de tamanho 8, ou seja, os índices a serem percorridos serão `[0, 1, 2, 3, 4, 5, 6, 7]`, o que pode ser feito usando um laço `for`.

No código a seguir, é possível fazer o percurso da *string* pelos índices com o auxílio da função `len()` e da função `range()`.

```
frase = 'a bola rola'

for indice in range(len(frase)):
    if frase[indice] == 'a':
        showInformation('Encontrei a letra a no texto!' )
```

O laço `for` vai percorrer todos os caracteres da variável `frase` e, em cada vez, vai verificar se o caractere da frase naquela posição do `indice` é igual à letra `'a'`. Se verdadeiro, vai mostrar a mensagem, assim como no laço anterior.

FOLHA DE ATIVIDADES

O Jogo da Forca é um jogo muito famoso. Com uma palavra escondida, o jogador deve adivinhar quais letras compõem a palavra misteriosa, fornecendo uma letra a cada rodada. Com base no seu conhecimento do jogo, crie um programa do Jogo da Forca em Python. Para facilitar o seu processo de construção do programa, você pode seguir alguns passos básicos, descritos a seguir.

Crie variáveis: Você pode usar uma variável para guardar a palavra e outra variável para guardar a palavra apenas com as letras que o jogador acertou (palavraEscondida). As letras que não foram acertadas devem ser escondidas com o caractere *sublinhado* (_), também chamado de *underscore*. Por exemplo, considerando que o jogador acertou as letras A, X e I, as variáveis teriam:

`palavra`	ABACAXI	`palavraEscondida`	A_A_AXI

Crie um contador: As chances do jogador não podem ser infinitas. Crie um contador para contabilizar as chances. Ele pode iniciar de zero e ir aumentando a cada rodada. Enquanto este contador for menor que o número máximo de chances que você escolher, o jogador poderá continuar dando palpites. Chegando a zero, o jogador perdeu o jogo. Você pode usar um laço `while` ou um laço `for` para controlar as rodadas.

Requisite a letra: Dentro do laço da rodada, exiba uma mensagem pedindo uma letra ao jogador. Utilize a função *requestString* para capturar a letra do teclado.

Verifique a letra: Dentro do laço da rodada, percorra a *string* letra por letra e verifique se a letra escolhida é equivalente a alguma letra em `palavra`. Para cada letra percorrida, faça uma das duas opções:

- Se a letra escolhida for igual à letra que está sendo percorrida em `palavra`, junte esta letra a uma variável *string* temporária;
- Se a letra escolhida não for igual à letra que está sendo percorrida em palavra, junte a letra em `palavraEscondida` que esteja na mesma posição à variável *string* temporária.

No final do percurso, a variável *string* temporária estará mais próxima da `palavra`. Então, faça `palavraEscondida` guardar o valor desta string temporária.

Isso fará com que, ao acertar todas as letras da `palavra`, você tenha a palavra correta guardada em `palavraEscondida`.

Critério de Parada: Pense um pouco como seria o critério de parada, quando o jogador esgotar suas chances ou quando ele ganhar. É possível adicionar uma condição ao laço `while` ou usar um comando `break` para interromper o laço `for`, faça de acordo com o tipo de laço escolhido anteriormente.

AULA 6 – A Droga da Obediência

Nesta aula, os estudantes farão um programa que implementa o método de criptografia do livro de Pedro Bandeira intitulado A Droga da Obediência.

Objetivos	Desenvolver programas em Python que trabalhem com entrada e saída de dados textuais.
	Aplicar o conceito de loops para automatizar tarefas repetitivas.
	Utilizar estruturas condicionais para tomar decisões diante do valor dos dados.
	Manipular strings em Python através de loops.
Conteúdo	Variáveis.
	Entrada e saída.
	Loops.
	Condicionais.
	Strings.
	Manipulação de strings.
	Criptografia.
Atividades	A droga da obediência: O Código Tenis-Polar e o Código dos Karas.
Resultados de Aprendizagem	PC2. Usar passos básicos na solução de problemas algorítmicos para projetar soluções (por exemplo, declaração e exploração de problemas, exemplos de instâncias, design, implementação de uma solução, teste, avaliação).
	PC4. Definir um algoritmo como uma sequência de instruções que pode ser processada por um computador.
	PC8. Representar dados de várias formas, incluindo texto, sons, imagens e números.
	PC15. Examinar as conexões entre os elementos da matemática e da ciência da computação, incluindo números binários, lógica, conjuntos e funções.
	PC16. Fornecer exemplos de aplicações interdisciplinares do pensamento computacional.
	PCP5. Reunir e manipular dados usando uma variedade de ferramentas digitais.
	PCP10. Demonstrar compreensão dos algoritmos e sua aplicação prática.

	PCP11. Implementar soluções de problemas usando uma linguagem de programação, incluindo: comportamento de looping, instruções condicionais, lógica, expressões, variáveis e funções.
Materiais	JES instalado. projeto1_unidade3_aula6.py. Folha de Atividades.

DESCRIÇÃO DAS ATIVIDADES

Nesta aula, os estudantes farão um programa que implementa o método de criptografia do livro de Pedro Bandeira intitulado *A droga da Obediência*.

1ª Parte: O professor deve iniciar a aula perguntando se alguém já leu o livro *A Droga da Obediência*. O professor deve então falar brevemente da história:

"Miguel é o líder do grupo "Karas", que desvendam crimes e mistérios. Além dele, o grupo é composto por Crânio, o "gêniozinho" da turma, Magrí, a melhor atleta da escola, Calu, o melhor ator e Chumbinho, o valente e esperto. Além de líder do grupo altamente secreto, Miguel é também presidente do Grêmio do Colégio Elite, um colégio democrático com um sistema de decisões em um conselho formado por alunos e professores. Miguel reúne o grupo em um esconderijo para falar sobre a onda de desaparecimentos que ocorrem em vários colégios, seguidamente, onde ao todo 27 estudantes desapareceram." – Disponível em: Wikipedia.

O professor deve apresentar a sinopse do livro e descrever os códigos de criptografia que os integrantes de grupo Karas desenvolveram para garantir a troca segura de mensagens entre eles.

Código Tenis-Polar: substitui as letras da palavra *tenis* pelas letras correspondentes da palavra *polar*. Assim:

```
T = P
E = O
N = L
I = A
S = R
```

O professor então deve mostrar o programa do Projeto1_unidade3_aula6.py, que implementa a codificação tenis-polar. Ele deve executar em sala, sem mostrar detalhes da implementação. O professor deve discutir com os alunos sobre como este programa pode ser desenvolvido seguindo a mesma lógica do programa criado na aula anterior, sobre criptografia de César.

2ª Parte: Os estudantes devem implementar um programa que implemente o Código dos Karas, também parte da história do livro *A Droga da Obediência*. A explicação desta criptografia está disponível na Folha de Atividades. O professor deve oferecer ajuda individual, quando solicitado. Os estudantes que finalizarem esta atividade antes dos colegas, podem implementar o Desafio 2, também disponível na Folha de Atividades.

3ª Parte: Ao final da aula, os estudantes devem brincar, trocando mensagens criptografadas entre si.

FOLHA DE ATIVIDADES

O livro *A Droga da Obediência* de Pedro Bandeira descreve um grupo de jovens, intitulado *Os Karas*, que investigavam e ajudavam a desvendar crimes. O grupo cria uma série de codificações para poder trocar mensagens em segurança durante as investigações.

Código Tenis-Polar:

Este código substitui as letras da palavra *tenis* pelas letras correspondentes da palavra *polar*. As letras da mensagem que não estão na palavra *tenis* permanecem iguais. Assim:

```
T = P
E = O
N = L
I = A
S = R
```

Exemplos:
CASA = CARA (Apenas o S está na palavra tenis)
ESCOLA = ORCOLA (Apenas E e S estão na palavra tenis)

Código dos Karas:

Este código substitui as vogais e mantém as consoantes. Assim:

```
A = AIS
E = ENTER
I = INIS
O = OMBER
U = UFTER
```

Exemplos:
CASA = CAISSAIS
ESCOLA = ENTERSCOMBERLAIS

Desafio 1 – Código dos Karas

Implemente um programa que codifique e decodifique mensagens com o Código dos Karas.

Desafio 2 – Combinando códigos

Implemente um programa que codifique e decodifique mensagens com a seguinte lógica:

Codificação: Transforma o texto seguindo o Tenis-Polar e, em seguida, transforma o texto seguindo o Código dos Karas.

Decodificação: Transforma o texto seguindo o Código dos Karas e, em seguida, transforma o texto seguindo o Tenis-Polar.

AULA 7 – Ovos da Páscoa

Nesta aula, os estudantes desenvolverão o código para encontrar uma mensagem escondida em um arquivo de texto através da linguagem Python.

Objetivos	Desenvolver programas em Python que trabalhem com entrada e saída de dados textuais. Aplicar o conceito de loops para automatizar tarefas repetitivas. Utilizar estruturas condicionais para verificação de caracteres. Implementar operações lógicas para verificação de mecanismos do jogo. Utilizar indexação de strings para identificação de caracteres separados. Desenvolver programas em Python com leitura de arquivos.
Conteúdo	Variáveis. Entrada e saída. Loops. Condicionais. Expressões aritméticas. Expressões lógicas. Strings. Indexação de strings. Arquivos.
Atividades	Ovo da Páscoa.
Resultados de Aprendizagem	PC2. Usar passos básicos na solução de problemas algorítmicos para projetar soluções (por exemplo, declaração e exploração de problemas, exemplos de instâncias, design, implementação de uma solução, teste, avaliação). PC4. Definir um algoritmo como uma sequência de instruções que pode ser processada por um computador. PC8. Representar dados de várias formas, incluindo texto, sons, imagens e números. PC15. Examinar as conexões entre os elementos da matemática e da ciência da computação, incluindo números binários, lógica, conjuntos e funções. PC16. Fornecer exemplos de aplicações interdisciplinares do pensamento computacional.

	PCP5. Reunir e manipular dados usando uma variedade de ferramentas digitais. PCP10. Demonstrar compreensão dos algoritmos e sua aplicação prática. PCP11. Implementar soluções de problemas usando uma linguagem de programação, incluindo: comportamento de looping, instruções condicionais, lógica, expressões, variáveis e funções. PCP15. Coletar e analisar dados que são produzidos a partir de várias execuções de um programa de computador.
Materiais	JES instalado. projeto1_unidade3_aula7.py. Folha de Atividades.

DESCRIÇÃO DAS ATIVIDADES

Nesta aula, os estudantes implementarão um código para identificar uma mensagem escondida em um arquivo de texto, disponibilizado pelo professor.

1ª Parte: O professor deve disponibilizar o arquivo em formato texto e a Folha de Atividades para os estudantes. Os estudantes irão identificar a mensagem escondida no texto, através da identificação de caracteres maiúsculos e espaços extras (mas isso não deve ser dito a eles). Essa etapa deve ser realizada sem o computador.

```
Esse texto tem Uma  menSagem escondida. O Unico  que pode
desvendar esse misterio e UM(a)  bravo(a) guErreiro(a),
eXtremamente habilidoso(a) e Curioso(a) sobrE os segredos
LENdarios da criptografia. esse guerreiro(a) deve dominar as
arTEs imortais da  PROGRAMAcao. nao sera facil, mas certamente
voce sera capaz de DescObRir a mensagem!
```

Figura 2 - Texto disponibilizado

2ª Parte: A percepção da mensagem deve ser realizada pelos estudantes, visando estimular a capacidade de identificar padrões. Após o reconhecimento do padrão da mensagem, os estudantes devem implementar em Python o código que leia um texto de um arquivo e exiba a mensagem escondida.

EU SOU UM EXCELENTE PROGRAMADOR!

Figura 3 - Frase escondida

FOLHA DE ATIVIDADES

Na Computação, o termo Ovos de Páscoa, ou *Easter Eggs*, é muito utilizado para designar mensagens escondidas em um programa de computador. Há muito tempo, alguns programadores esconderam os seus nomes dentro dos jogos de console. Esse conteúdo só poderia ser acessado se os jogadores realizassem alguns passos definidos. Hoje em dia, empresas como a Microsoft, Facebook e Google possuem vários *Easter Eggs* para divertir os usuários. Por exemplo, se digitarmos "*Do a Barrel Roll*" no buscador da Google vamos ter uma surpresa!

Nesta atividade, há uma mensagem escondida em um arquivo de texto. Analise o texto e verifique o que a mensagem esconde. Em seguida, crie uma estratégia, baseada no padrão que você descobriu, para que o seu programa possa exibir a mensagem.

```
Esse texto tem Uma  menSagem escondida. O Unico  que pode
desvendar esse misterio e UM(a)  bravo(a) guErreiro(a),
eXtremamente habilidoso(a) e Curioso(a) sobrE os segredos
LENdarios da criptografia. esse guerreiro(a) deve dominar as
arTEs imortais da  PROGRAMAcao. nao sera facil, mas certamente
voce sera capaz de DescObRir a mensagem!
```

Abrindo o arquivo: Para abrir o arquivo de texto, você pode utilizar a função *open*. Esta função permite abrir um arquivo que está no seu computador, bastando passar para a função o caminho do arquivo, ou seja, em qual pasta o arquivo está no seu computador e o nome dele. O arquivo pode ser aberto de diversas formas, para leitura (modo *read*) ou para escrita (*write*). Neste caso, vamos apenas ler a mensagem, então, usaremos o modo de leitura (*read*), identificado pela letra 'r', na função open. Após a execução da função, todo o conteúdo do arquivo estará na variável texto.

```
texto = open('C:/Usuário/Desktop/texto_unidade3_aula7.txt','r')
```

Percorrendo o arquivo: Para percorrer o arquivo, nós podemos utilizar dois laços, um dentro do outro (também chamados de laços aninhados). O primeiro, para percorrer linha por linha. É possível exibir a linha inteira. O segundo, para percorrer cada letra de cada linha. Caso o arquivo só tenha uma linha, não precisaremos de dois laços. O código a seguir exibe o conteúdo da linha e depois cada letra desta linha.

```
for linha in texto:
  print(linha)
  for letra in linha:
    print(letra)
```

Identificando a mensagem: Agora você está pronto para implementar o seu código. Utilize suas habilidades com condicionais, loops e strings para exibir a mensagem.

AULA 8 – Jogo da Velha

Nesta aula, os estudantes farão um programa que implementa o Jogo da Velha. No jogo criado pelos estudantes, o jogador enfrenta o computador.

Objetivos	Desenvolver programas em Python que trabalhem com entrada e saída de dados textuais.
	Aplicar o conceito de loops para automatizar tarefas repetitivas.
	Utilizar estruturas condicionais para tomar decisões diante do valor dos dados.
	Utilizar listas para agregar, armazenar e consultar dados.
	Utilizar funções para segmentar as rotinas de seu programa, organizando o código de maneira mais concisa.
Conteúdo	Variáveis.
	Entrada e saída.
	Loops.
	Condicionais.
	Strings.
	Listas e sua indexação.
	Funções.
Atividades	Jogo da Velha.
Resultados de Aprendizagem	PC2. Usar passos básicos na solução de problemas algorítmicos para projetar soluções (por exemplo, declaração e exploração de problemas, exemplos de instâncias, design, implementação de uma solução, teste, avaliação).
	PC4. Definir um algoritmo como uma sequência de instruções que pode ser processada por um computador.
	PC8. Representar dados de várias formas, incluindo texto, sons, imagens e números.
	PC15. Examinar as conexões entre os elementos da matemática e da ciência da computação, incluindo números binários, lógica, conjuntos e funções.
	PC16. Fornecer exemplos de aplicações interdisciplinares do pensamento computacional.
	PCP5. Reunir e manipular dados usando uma variedade de ferramentas digitais.
	PCP10. Demonstrar compreensão dos algoritmos e sua aplicação prática.

	PCP11. Implementar soluções de problemas usando uma linguagem de programação, incluindo: comportamento de looping, instruções condicionais, lógica, expressões, variáveis e funções.
Materiais	JES instalado. projeto1_unidade3_aula8.py. Folha de Atividades.

DESCRIÇÃO DAS ATIVIDADES

Nesta aula, os estudantes serão desafiados a implementarem o Jogo da Velha, onde o jogador enfrenta o computador.

1ª Parte: O professor deve iniciar a aula mostrando o programa do Projeto1_unidade3_aula8.py, que consiste em um Jogo da Velha onde o jogador enfrenta o computador. Ele deve executar em sala, sem mostrar detalhes da implementação. Alguns estudantes podem ser escolhidos, ou se voluntariarem, para jogar. O professor deve discutir com os alunos como este jogo pode ser implementado. A discussão deve culminar em uma proposta de algoritmo para o jogo, que inclui o comportamento inteligente do computador. O professor, conhecendo detalhes de implementação, deve conduzir a discussão para obter um algoritmo similar ao programa dado como exemplo.

2ª Parte: Os estudantes devem implementar o Jogo da Velha, seguindo os desafios da Folha de Atividades. O professor deve oferecer ajuda individual, quando solicitado.

3ª Parte: Ao final da aula, os estudantes devem brincar com o jogo criado.

FOLHA DE ATIVIDADES

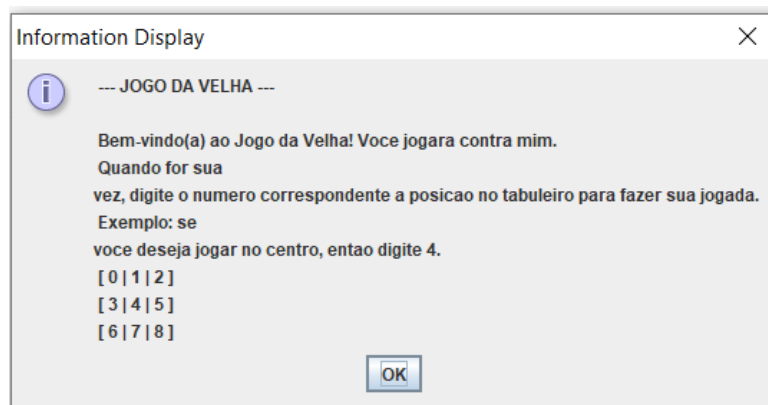
Na aula de hoje, você vai implementar um Jogo da Velha inteligente, onde o jogador enfrenta o computador! Siga os desafios propostos:

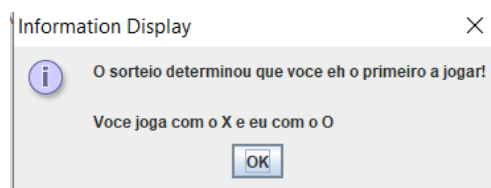
Desafio 1 – Definindo variáveis necessárias

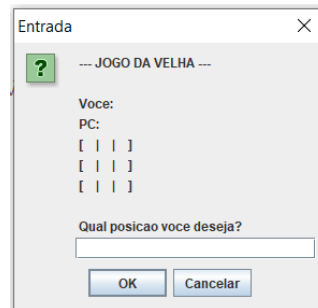
Você vai precisar de algumas variáveis para armazenar:

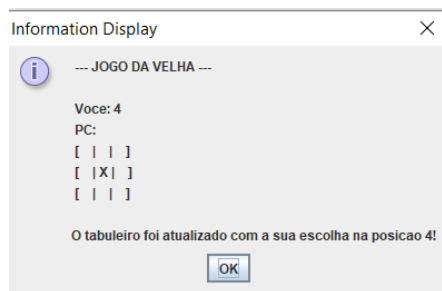
- Quantas jogadas o jogo já teve;
- O tabuleiro com as nove posições do jogo;
- Quem é o jogador da vez;
- As jogadas do jogador humano;
- As jogadas do computador;
- O vencedor.

Desafio 2 – Apresentando o Tabuleiro

Crie a tela de apresentação do seu jogo. Você pode informar as instruções principais:

Desafio 3 – Quem joga primeiro?

Defina quem é o primeiro jogador, o humano ou o computador. Use uma variável específica para armazenar quem é o jogador da vez. Informe ao usuário quem é o primeiro jogador.

Desafio 4 – O que acontece quando o jogo está na vez do jogador humano?

O jogador humano deve escolher uma posição livre no tabuleiro. Para isso, crie uma tela em que o usuário visualize o estado atual do tabuleiro e insira a posição em que deseja jogar.

Ao inserir a mensagem, o programa deve:

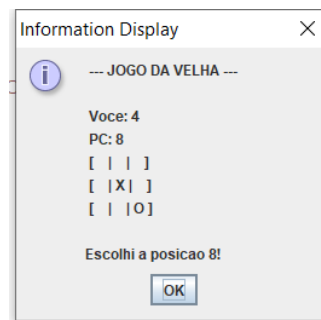
- Marcar na variável específica a posição do tabuleiro escolhida;
- Adicionar a posição à lista de jogadas do jogador humano;
- Incrementar o número de jogadas;
- Determinar que é a vez do jogador computador;
- Apresentar o tabuleiro atualizado.

Desafio 4 – O que acontece quando está na vez do jogador PC?

O jogador PC ou seja, o computador, deve fazer uma jogada. Sinta-se livre para elaborar uma jogada mais sofisticada, avaliando as chances de vitória, por exemplo. Após definir a posição de jogada, o programa deve:

- Marcar na variável específica a posição do tabuleiro escolhida;
- Adicionar a posição à lista de jogadas do jogador humano;
- Incrementar o número de jogadas;
- Determinar que é a vez do jogador humano;
- Apresentar o tabuleiro atualizado.

Dica: Uma das ideias mais simples é se o computador simplesmente sortear uma posição livre no tabuleiro. O programa deve sortear um número e verificar se ele corresponde a uma posição livre no tabuleiro. Se sim, prossiga, senão, você deve sortear outro número. Para isso, você pode utilizar uma função de sorteio de números aleatórios chamada `randint`.

```
escolha = randint(0,8)
while  tabuleiro[escolha] == 'X' or tabuleiro[escolha] == 'O' :
     escolha = randint(0,8)
```

Depois do sorteio é importante informar a escolha ao jogador PC e mostrar o tabuleiro atualizado.

Desafio 5 – Checando o vencedor

Sempre que uma jogada for feita, você deve verificar se houve vitória. Considerando que o tabuleiro possui 9 posições, que podem ser acessadas pelos índices 0 a 8 da lista, estas são as opções de vitória:

0	1	2

3	4	5

6	7	8

0		
	4	
		8

0		
3		
6		

	1	
	4	
	7	

		2
		5
		8

		2
	4	
6		

Ao todo, oito possibilidades devem ser verificadas. Você deve verificar se o jogador humano ganhou, se o jogador PC ganhou ou se a quantidade de jogadas ultrapassou 9 (então ninguém ganhou). Você deve utilizar o resultado dessas verificações para continuar o jogo, repetindo as jogadas, ou parar o jogo e informar o resultado final.

UNIDADE IV

Arduino

AULA 1 - Construindo um Semáforo

Nesta aula, os estudantes serão apresentados ao universo da eletrônica com Arduino. Através de três projetos simples, os estudantes aprenderão como programar a placa Arduino para incluir comportamentos específicos em circuitos com LEDs.

Objetivos	Entender a funcionalidade dos seguintes componentes: protoboard, LED, Arduino, jumper, suporte para pilhas e resistor.
	Compreender o esquema de montagem da protoboard.
	Compreender que a introdução do Arduino em um circuito torna possível estabelecer um comportamento programado no circuito estabelecido.
	Compreender que circuitos programados possuem estados diferentes.
Conteúdo	Circuitos elétricos e eletrônicos.
	Arduino.
	Programação: sequências e loops.
	Comandos do mBlock: eventos, controle e Pin.
Atividades	Projeto 1 – Ligando um LED.
	Projeto 2 – Ligando três LEDs.
	Projeto 3 – Construindo um semáforo.
Resultados de Aprendizagem	PC2. Usar passos básicos na solução de problemas algorítmicos para projetar soluções (por exemplo, declaração e exploração de problemas, exemplos de instâncias, design, implementação de uma solução, teste, avaliação).
	PC7. Descrever e analisar uma sequência de instruções que estão sendo seguidas (por exemplo, descrever o comportamento de um personagem em um videogame conforme orientado por regras e algoritmos).
	PCP10. Demonstrar compreensão dos algoritmos e sua aplicação prática.
	PCP11. Implementar soluções de problemas usando uma linguagem de programação, incluindo: comportamento de looping, instruções condicionais, lógica, expressões, variáveis e funções.
	PCP13. Identificar carreiras interdisciplinares que são aprimoradas pela ciência da computação.
	C7. Colaborar com colegas, especialistas e outras pessoas usando práticas colaborativas, como programação em pares, trabalho em equipes de projeto e participação em atividades de aprendizado ativo em grupo.

	C8. Apresentar as disposições necessárias para colaboração: fornecer feedback útil, integrar feedback, compreender e aceitar múltiplas perspectivas, socialização.
	CDC1. Reconhecer que os computadores modelam o comportamento inteligente (como encontrado em robótica, reconhecimento de fala e linguagem e animação por computador).
	CDC3. Reconhecer que os computadores são dispositivos que executam programas. CDC4. Identificar uma variedade de dispositivos eletrônicos que contêm processadores computacionais.
	CDC9. Descrever o que distingue os seres humanos das máquinas, concentrando-se na inteligência humana versus a inteligência das máquinas e nas maneiras pelas quais podemos comunicar.
	CDC10. Descrever maneiras pelas quais os computadores usam modelos de comportamento inteligente (por exemplo, movimento do robô, compreensão da fala e da linguagem, e visão computacional).
Materiais	Folha de atividades.
	Kit de eletrônica.
	mBlock.

DESCRIÇÃO DAS ATIVIDADES

Nesta aula, os estudantes serão apresentados ao universo da eletrônica com Arduino. Serão implementados três projetos. O primeiro projeto tem como objetivo ligar um LED; o segundo projeto, ligar três LEDs; e o último projeto é um protótipo de semáforo controlado pela placa Arduino.

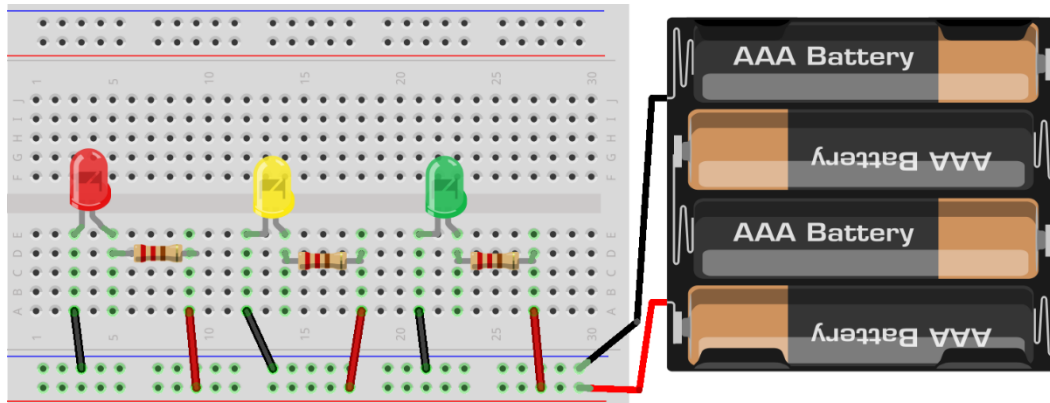
1ª Parte: O professor deve iniciar a aula perguntando aos estudantes se eles já tiveram a curiosidade de abrir um aparelho eletrônico velho, como um rádio, ou uma TV. O professor deve apresentar o fato de que estes aparelhos possuem placas de circuitos eletrônicos, que podem ter algum tipo de programação. O professor também deve lembrar aos estudantes a experiência prévia deles com o robô mBot, que nada mais é do que um circuito com diversos módulos como motores e sensores que podem ser programados. O professor deve então apresentar aos estudantes a placa Arduino, que é uma "plataforma" de hardware e software que permite a criação de protótipos tecnológicos, e é muito utilizado por estudantes, engenheiros, inventores, dentre outras profissões que trabalham com a criação de tecnologia.

2ª Parte: O professor deve apresentar os componentes necessários para a implementação dos projetos da aula e descrever brevemente a função de cada um deles no circuito. Esta parte da aula pode ter como base a seção de Tópicos Relevantes. O professor pode retirar o fundo de uma protoboard velha, se disponível, para demonstrar como os furos da protoboard estão interligados através de material condutor. Ao tratar dos resistores, não é necessário falar sobre o reconhecimento dos resistores através dos padrões de linhas coloridas, mas o professor pode falar sobre isso, caso julgue necessário.

A medida em que o professor apresenta os componentes, é importante que os grupos manipulem estes componentes, aprendendo a identificá-los. Os estudantes devem ser divididos de acordo com a quantidade dos materiais disponíveis. O ideal é que os estudantes trabalhem em dupla. Não recomendamos grupos com mais do que quatro integrantes.

3ª Parte: O primeiro circuito tem como objetivo ligar um LED, alimentando a *protoboard* através das pilhas AA. Essa atividade tem como base o esquemático do Projeto 1 da Folha de Atividades. O professor deve ensinar aos estudantes como interpretar o esquemático e montar o primeiro circuito passo a passo. Cada grupo deve montar o seu circuito.

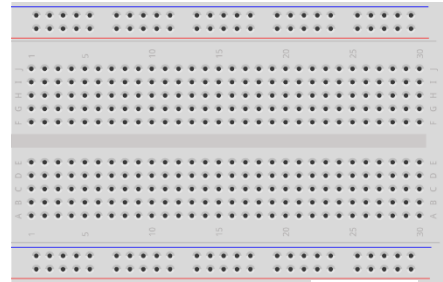
No processo de montagem deste circuito, é interessante que o professor fale da corrente elétrica e da importância de criar um caminho fechado para a corrente elétrica. O professor pode falar que a corrente elétrica possui um sentido real e um sentido convencional. O sentido real é o movimento dos elétrons e acontece do polo negativo para o polo positivo. O sentido convencional corresponde ao sentido do campo elétrico no interior do condutor e vai do polo positivo para o negativo.

4ª Parte: O segundo circuito tem como objetivo ligar três LEDs, com as cores vermelho, amarelo e verde. O circuito deve ser alimentado através das pilhas AA. Esta atividade tem como base o Projeto 2 da Folha de Atividades, que não possui esquemático. O professor deve desafiar os estudantes a implementarem este circuito.

Após a finalização da montagem, o professor deve perguntar a turma o que fazer para que este circuito se transforme em um semáforo, explanando posteriormente sobre a necessidade de programar o comportamento do circuito. O professor também deve falar sobre os estados de um semáforo: Pare, avance e atenção.

5ª Parte: O terceiro circuito tem como objetivo construir o protótipo de um semáforo, com as cores vermelho, amarelo e verde. Esta atividade tem como base o Projeto 3 da Folha de Atividades. O professor deve desafiar os estudantes a implementarem este circuito, com base no esquemático disponibilizado na Folha de Atividades. Os LEDs devem ser conectados as portas digitais do Arduino, que podem ou não emitir corrente elétrica, de acordo com a programação recebida. À medida em que os grupos finalizem a montagem, devem ser direcionados para programar este circuito no mBlock, utilizando as dicas oferecidas pelo projeto. O professor pode dar orientações gerais à turma sobre como selecionar o Arduino dentro do mBlock e como embarcar o projeto, mas é interessante que ele não discorra sobre a programação do circuito.

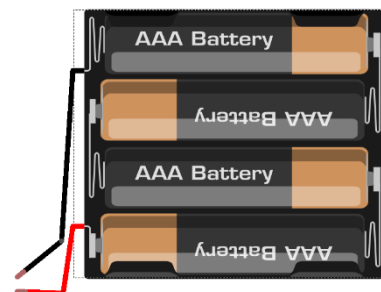
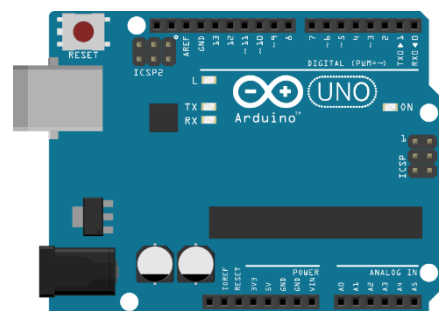
TÓPICOS RELEVANTES

Nesta unidade vamos manipular diversos componentes durante a montagem dos projetos com Arduino. Os componentes utilizados para a montagem dos projetos da aula de hoje são:

- **Protoboard:** é uma placa que serve como base para a montagem dos circuitos. Possui uma lógica especifica para estabelecer as conexões.

- **LED:** diodo emissor de luz (em inglês: *light-emitting diode*). Componente utilizado para a emissão de luz. Possui polaridade, que pode ser identificada do seguinte modo: terminal negativo é o menor terminal.

- **Jumper:** fio utilizado para estabelecer conexões entre os componentes. Podem ser do tipo macho/macho, quando tem as duas extremidades pontiagudas, macho/fêmea, quando tem apenas uma das extremidades pontiagudas ou fêmea/fêmea, quanto nenhum das extremidades é pontiaguda.

- **Resistor:** Componente que pode ser utilizado com a finalidade de transformar energia elétrica em energia térmica, por meio do efeito Joule, ou para limitar a corrente elétrica em um circuito. Em nossos projetos, o resistor tem como função limitar a corrente elétrica e utilizaremos um modelo 220R sempre associado aos LEDs.

- **Suporte para pilhas:** é um suporte onde pilhas AA podem ser encaixadas e combinadas para obter uma determinada voltagem. Cada pilha AA tem 1,5V e nossos suportes acomodam 4 pilhas em série, ou seja, 6V.

- **Arduino:** Placa de circuito que possui um circuito integrado interno e diversos módulos, podendo ser programada para a criação de protótipos.

FOLHA DE ATIVIDADES

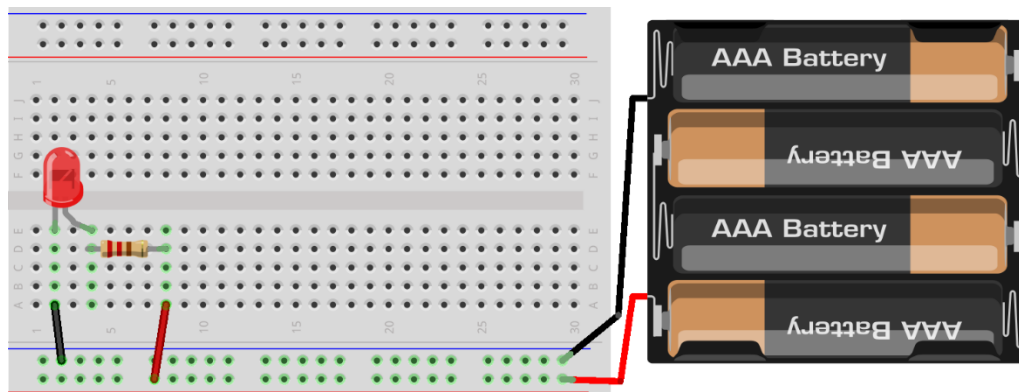
Seja bem-vindo ao universo da eletrônica com Arduino! Nesta unidade, você irá trabalhar um bocado com esta placa que é amplamente utilizada para a criação de protótipos tecnológicos. Sabe como é, né? Você já é praticamente um engenheiro criador de tecnologias! Na aula de hoje, vamos implementar três projetos.

Projeto 1 – Ligando um LED

OBJETIVO: Acender um LED, alimentando a *protoboard* com quatro pilhas AA de 1,5 V.

COMPONENTES NECESSÁRIOS:

Quantidade	Item
1	LED (com a cor que você desejar)
1	Resistor 220R
2	Jumpers do tipo macho/macho
1	Suporte para quatro pilhas AA
4	Pilhas AA 1,5 V

ESQUEMÁTICO:

Projeto 2 – Ligando Três LEDs

OBJETIVO: Acender três LEDs, com as cores vermelho, amarelo e verde. O circuito deve ser alimentado por quatro pilhas AA de 1,5 V.

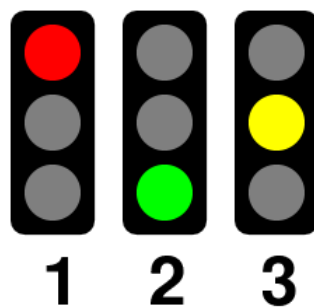
COMPONENTES NECESSÁRIOS:

Quantidade	Item
3	LED (cores: vermelho, amarelo e verde)
3	Resistor 220R
6	Jumpers do tipo macho/macho
1	Suporte para quatro pilhas AA
4	Pilhas AA 1,5 V

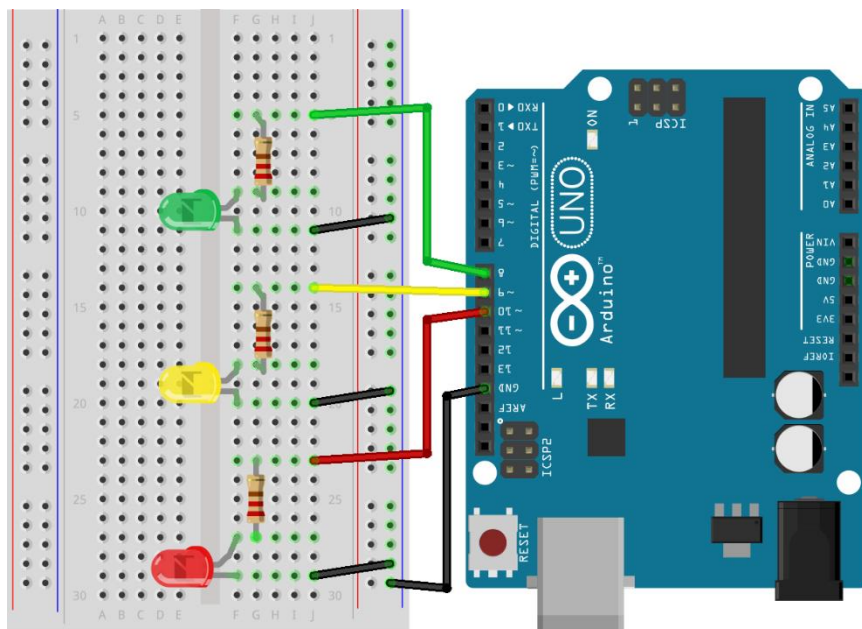
ESQUEMÁTICO: Como você tem se mostrado um engenheiro supertalentoso, não vamos oferecer o esquemático desta vez. Utilize a mesma lógica do circuito do Projeto 1!

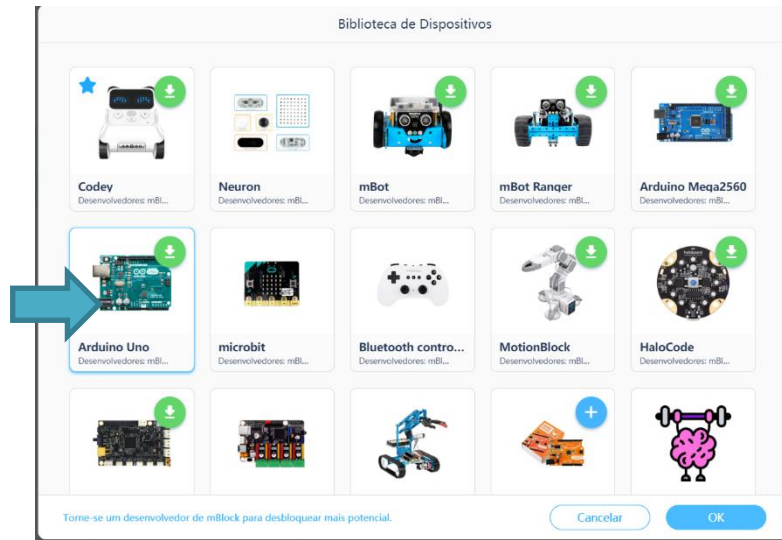
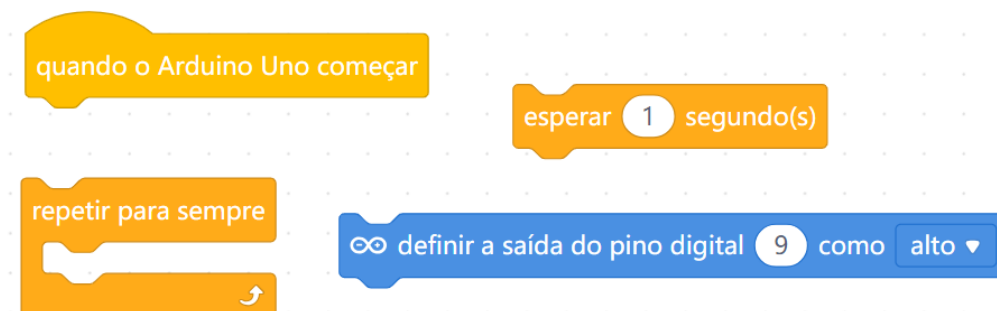
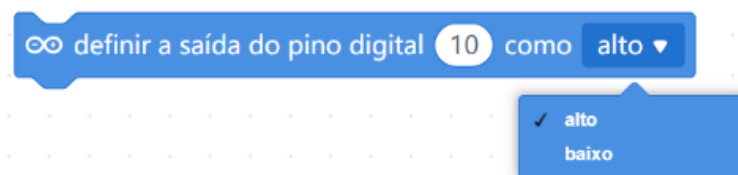
Projeto 3 – Construindo um Semáforo

OBJETIVO: Construir um circuito que tenha o comportamento de um semáforo.

REQUISITOS: O circuito deve ter três LEDs, com as cores vermelho, amarelo e verde. O circuito tem três estados: deve passar um tempo no sinal vermelho, depois deve passar um tempo no sinal verde e, por fim, um tempo menor no sinal amarelo. Este processo de mudança de estados deve se repetir sempre!

COMPONENTES NECESSÁRIOS:

Quantidade	Item
1	Arduino Uno
3	LED (cores: vermelho, amarelo e verde)
3	Resistor 220R
6	Jumpers do tipo macho/macho

ESQUEMÁTICO:

PROGRAMAÇÃO NO MBLOCK: Para criar um circuito com comportamentos bem definidos, você deve utilizar o Arduino e programá-lo de acordo com o objetivo do circuito. O Arduino Uno pode ser programado através do mBlock (você lembra do mBlock, né?). Como você é um programador muito experiente, não é necessário que apresentemos o programa inteiro, mas vamos te dar algumas dicas.

1ª Dica: Ao abrir o mBlock, remova o dispositivo padrão "Codey" e adicione o dispositivo Arduino Uno.

2ª Dica: Para programar o Arduino, você deve utilizar os comandos abaixo. Pense num algoritmo para o seu programa.

O comando *Definir saída do pino digital...* é o mais importante! Nele, você deve especificar a porta digital do Arduino onde o componente está conectado e se, através dessa porta, passará ou não corrente. Por exemplo, o LED vermelho está conectado à porta digital 10 do Arduino. Para programá-lo, vamos utilizar este comando, passando 10 como parâmetro. Para ligar o LED, devemos escolher a opção "alto" e, para desligar o LED, devemos escolher a opção "baixo".

AULA 2 – Medindo a Luminosidade em um Ambiente

Nesta aula, os estudantes serão apresentados ao sensor de luminosidade LDR e irão criar um circuito capaz de medir a luminosidade em um ambiente.

Objetivos	Entender a funcionalidade do sensor LDR e como ele pode ser incorporado num circuito eletrônico com a placa Arduino. Compreender a diferença entre um sinal digital e um sinal analógico e como eles são interpretados pelos computadores. Criar um circuito que meça o nível de luminosidade do ambiente.
Conteúdo	Circuitos elétricos e eletrônicos. Arduino. Sensor de luminosidade LDR. Sinais digitais e analógicos. Programação: sequências, loops e condicionais. Comandos do mBlock: eventos, controle, operadores, variáveis e Pin.
Atividades	Identificando a tensão lida. Medindo a Luminosidade.
Resultados de Aprendizagem	PC2. Usar passos básicos na solução de problemas algorítmicos para projetar soluções (por exemplo, declaração e exploração de problemas, exemplos de instâncias, design, implementação de uma solução, teste, avaliação). PC7. Descrever e analisar uma sequência de instruções que estão sendo seguidas (por exemplo, descrever o comportamento de um personagem em um videogame conforme orientado por regras e algoritmos). PCP10. Demonstrar compreensão dos algoritmos e sua aplicação prática. PCP11. Implementar soluções de problemas usando uma linguagem de programação, incluindo: comportamento de looping, instruções condicionais, lógica, expressões, variáveis e funções. PCP13. Identificar carreiras interdisciplinares que são aprimoradas pela ciência da computação. C7. Colaborar com colegas, especialistas e outras pessoas usando práticas colaborativas, como programação em pares, trabalho em equipes de projeto e participação em atividades de aprendizado ativo em grupo. C8. Apresentar as disposições necessárias para colaboração: fornecer feedback útil, integrar feedback, compreender e aceitar múltiplas perspectivas, socialização.

	CDC1. Reconhecer que os computadores modelam o comportamento inteligente (como encontrado em robótica, reconhecimento de fala e linguagem e animação por computador).
	CDC3. Reconhecer que os computadores são dispositivos que executam programas. CDC4. Identificar uma variedade de dispositivos eletrônicos que contêm processadores computacionais.
	CDC9. Descrever o que distingue os seres humanos das máquinas, concentrando-se na inteligência humana versus a inteligência das máquinas e nas maneiras pelas quais podemos comunicar.
	CDC10. Descrever maneiras pelas quais os computadores usam modelos de comportamento inteligente (por exemplo, movimento do robô, compreensão da fala e da linguagem, e visão computacional).
Materiais	Folha de atividades.
	Kit de eletrônica.
	mBlock.

DESCRIÇÃO DAS ATIVIDADES

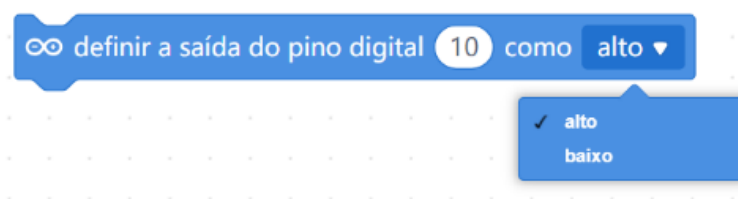
Nesta aula, os estudantes serão apresentados ao sensor de luminosidade LDR e irão criar um circuito capaz de medir a luminosidade em um ambiente. Os estudantes já trabalharam com a medição de luminosidade através do sensor embarcado no robô mBot. Nesta aula, os estudantes devem trabalhar novamente com este sensor mas em um nível maior de profundidade.

1ª Parte: O professor deve iniciar a aula perguntando aos estudantes se eles lembram que o robô mBot é capaz de medir luminosidade. Então, deve lembrá-los da aula do livro do 7º ano onde os estudantes manipularam o sensor de luminosidade embarcado no robô mBot.

O sensor de luminosidade do mBot é um componente chamado *Light Dependent Resistor* – LDR. Traduzido para o português, é um resistor dependente de luz. Este componente é capaz de medir a luminosidade do ambiente, retornando para a placa do mBot um valor numérico. O sensor de luz LDR, quando lido pelo Arduino, retorna um valor entre 0 e 1023. Quanto maior o valor retornado, mais claro está o ambiente.

Ao criar programas que utilizam este sensor, devem ser estabelecidos valores-limite. Por exemplo, uma leitura do sensor inferior a 500 representa um ambiente escuro, enquanto uma leitura do sensor maior que 500 representa um ambiente mais claro. Em outro exemplo, uma leitura do sensor inferior a 300, representa um ambiente mais escuro do que uma leitura do sensor que está entre 300 e 500.

2ª Parte: O professor deve apresentar o LDR e explicar que a lógica de utilização é a mesma. Porém, ao utilizar o Arduino, devemos ter um conhecimento maior sobre como este sensor funciona. O ponto mais importante é que este sensor é analógico e deve ser conectado à uma das portas analógicas do Arduino.

Um pino digital do Arduino, trabalha apenas com o valor 0 ou 1. Ao especificar “alto”, estamos estabelecendo que pino estará sujeito a uma tensão de 5V e, ao especificar “baixo”, estamos dizendo que, naquele pino, não haverá tensão, ou seja, terá uma tensão de 0V.

O Arduino Uno possui 6 pinos analógicos. A diferença entre um pino analógico e um digital, é que o analógico pode assumir um valor em uma faixa de valores numéricos. Para medir o valor de sinais analógicos, o Arduino utiliza de um conversor analógico-digital (ADC) soldado à sua placa. Este conversor transforma a tensão, que é analógica, num valor digital. O mBlock facilita o modo

como interpretamos o valor convertido. O fato é que um valor da tensão, num pino de entrada analógica, é convertido em um valor de 0 a 1023. Se o valor lido for 0, então a tensão na entrada do pino é 0V, mas se o valor lido for 1023, então a tensão lida é 5V. Para fazer a leitura de um pino analógico, basta utilizar o comando *ler pino analógico*, informando em qual porta analógica o componente está conectado:

ler pino analógico (A) 0

3ª Parte: O professor deve continuar sua explanação sobre a conversão de valores analógicos em digitais. É importante fazer exemplos como:

Qual valor será lido pela porta analógica do Arduino, se ele receber uma entrada de 3V?
Para resolver esta questão basta fazer uma regra de três:

$$\frac{x}{1023} = \frac{3}{5} \rightarrow x = \frac{3 \times 1023}{5} = 613{,}8$$

Mas o mBlock não apresenta valores fracionários, aproximando o número 613,8 para o inteiro mais próximo, 614. Os estudantes devem fazer a primeira parte da Folha de Atividades. O professor deve reservar um tempo para que todos façam a atividade e, então, corrigir no quadro.

4ª Parte: Os estudantes devem fazer a segunda parte da Folha de Atividades, que é o Projeto 1 – Medindo a Luminosidade. O professor deve explicar brevemente do que se trata este projeto. Os estudantes devem ser divididos de acordo com a quantidade dos materiais disponíveis. O ideal é que os estudantes trabalhem em dupla. Não recomendamos grupos com mais do que quatro integrantes. Cada grupo deve montar e programar seu circuito seguindo o esquemático disponibilizado. O professor deve oferecer suporte à medida em que as dúvidas apareçam.

FOLHA DE ATIVIDADES

Continuando a nossa aventura pelo universo da Eletrônica, hoje é dia de medir luminosidade através do sensor chamado *Light Dependent Resistor* – LDR, um resistor dependente de luz.

Calculando o resultado da conversão Analógico-Digital!

Na aula anterior, ao trabalhar com LEDs, utilizamos as portas digitais do Arduino. Portas digitais trabalham com apenas dois valores de sinais: 0V ou 5V. Ou seja, ou tem sinal ou não tem sinal naquele pino! O sensor LDR, que é tema de estudo da aula de hoje, é um sensor analógico.

Para utilizar o LDR, precisamos conectá-lo a uma porta analógica do Arduino. Apesar de só trabalhar internamente com sinais digitais, o Arduino utiliza de um conversor analógico-digital que transforma a tensão, que é analógica, num valor digital. Um valor da tensão num pino de entrada analógica é convertido em um valor de 0 a 1023. Se o valor lido for 0, então a tensão na entrada do pino é 0V, mas se o valor lido for 1023, então a tensão lida é 5V. Como você é fera na matemática, responda as questões a seguir:

Qual valor será lido pela porta analógica do Arduino, se ele receber uma tensão de entrada de 1V?

____

Qual valor será lido pela porta analógica do Arduino, se ele receber uma tensão de entrada de 4V?

____

Qual valor será lido pela porta analógica do Arduino, se ele receber uma tensão de entrada de 2,5V?

____

Projeto 1 – Medindo a Luminosidade

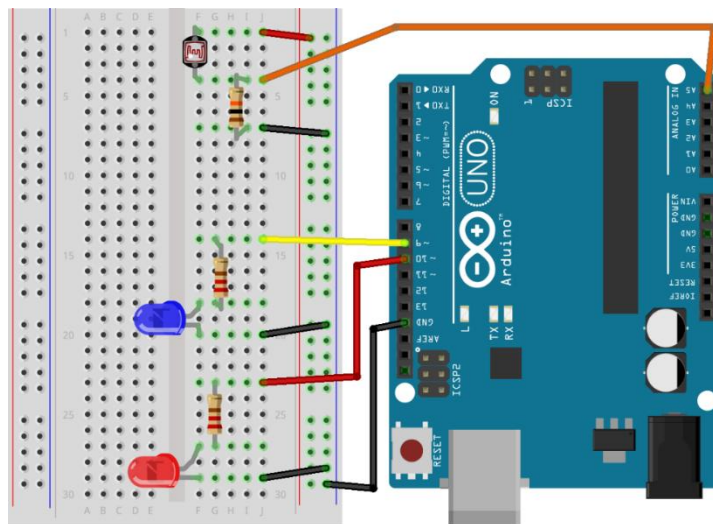
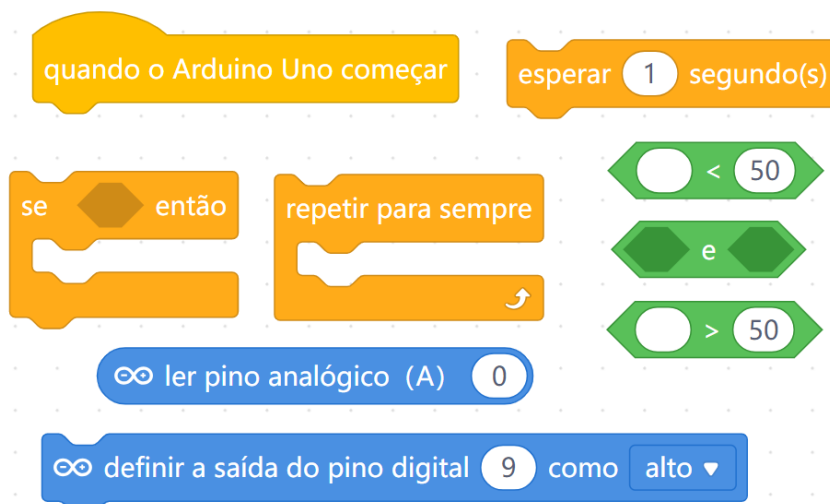
OBJETIVO: Construir um circuito capaz de medir a luminosidade em um ambiente.

REQUISITOS: Este circuito terá dois LEDs, vermelho e azul, e um sensor de luminosidade LDR. Ao medir a luminosidade do ambiente, devem ser identificados 3 níveis de luminosidade: escuro, parcialmente escuro e claro. Ao detectar o nível escuro, apenas o LED vermelho deve ser ligado. Ao identificar o nível parcialmente escuro, apenas o LED azul deve ser ligado. Ao identificar o nível claro, os dois LEDs devem permanecer apagados. O valor de entrada para o nível escuro é entre

0 e 2,5V; o valor de entrada para o nível parcialmente escuro é entre 2,5 e 4V; e o valor para o nível claro é acima de 4V.

COMPONENTES NECESSÁRIOS:

Quantidade	Item
1	Arduino Uno
1	LED (com a cor que você desejar)
1	Resistor 220R
1	Sensor LDR
1	Resistor de 10K
8	Jumpers do tipo macho/macho

ESQUEMÁTICO:

PROGRAMAÇÃO NO MBLOCK: Para criar o programa do circuito que mede luminosidade, você deve utilizar os comandos abaixo, podendo usar outros blocos caso ache necessário. Como você é um programador expert, deve pensar num algoritmo que faça a leitura do valor do sensor LDR, identifique em qual nível de luminosidade o valor lido se encaixa, e definir as ações correspondentes, de acordo com o objetivo definido para o circuito.

AULA 3 – Criando um Piano com o Arduino

Nesta aula, os estudantes desenvolverão o projeto de um Piano com o Arduino, utilizando um buzzer, para emitir sons, e push buttons, que funcionam como botões.

Objetivos	Entender a funcionalidade do push button e como incorporá-lo em um circuito eletrônico.
	Entender a funcionalidade do buzzer e como incorporá-lo em um circuito eletrônico.
	Criar um circuito eletrônico que defina comportamento de componentes como LEDs e Buzzers a partir da interação do usuário por meio de push buttons.
Conteúdo	Circuitos eletrônicos.
	Arduino.
	Push button.
	Buzzer.
	Programação: sequências, loops e condicionais.
	Comandos do mBlock: Eventos, Controle, Operadores e Pin.
Atividades	Projeto 1 – Piano Arduino.
	Projeto 2 – MiniPiano Arduino (Para os mais espertos).
Resultados de Aprendizagem	PC2. Usar passos básicos na solução de problemas algorítmicos para projetar soluções (por exemplo, declaração e exploração de problemas, exemplos de instâncias, design, implementação de uma solução, teste, avaliação).
	PC7. Descrever e analisar uma sequência de instruções que estão sendo seguidas (por exemplo, descrever o comportamento de um personagem em um videogame conforme orientado por regras e algoritmos).
	PCP10. Demonstrar compreensão dos algoritmos e sua aplicação prática.
	PCP11. Implementar soluções de problemas usando uma linguagem de programação, incluindo: comportamento de looping, instruções condicionais, lógica, expressões, variáveis e funções.
	PCP13. Identificar carreiras interdisciplinares que são aprimoradas pela ciência da computação.
	C7. Colaborar com colegas, especialistas e outras pessoas usando práticas colaborativas, como programação em pares, trabalho em equipes de projeto e participação em atividades de aprendizado ativo em grupo.

	C8. Apresentar as disposições necessárias para colaboração: fornecer feedback útil, integrar feedback, compreender e aceitar múltiplas perspectivas, socialização.
	CDC1. Reconhecer que os computadores modelam o comportamento inteligente (como encontrado em robótica, reconhecimento de fala e linguagem e animação por computador).
	CDC3. Reconhecer que os computadores são dispositivos que executam programas. CDC4. Identificar uma variedade de dispositivos eletrônicos que contêm processadores computacionais.
	CDC9. Descrever o que distingue os seres humanos das máquinas, concentrando-se na inteligência humana versus a inteligência das máquinas e nas maneiras pelas quais podemos comunicar.
	CDC10. Descrever maneiras pelas quais os computadores usam modelos de comportamento inteligente (por exemplo, movimento do robô, compreensão da fala e da linguagem, e visão computacional).
Materiais	Folha de atividades.
	Kit de eletrônica.
	mBlock.

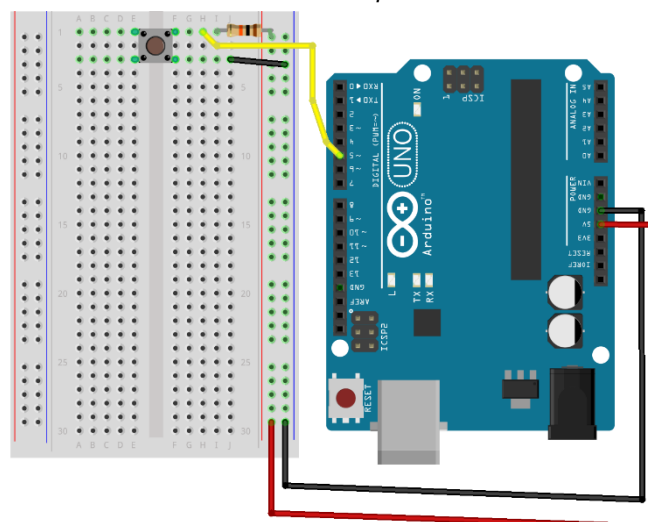
DESCRIÇÃO DAS ATIVIDADES

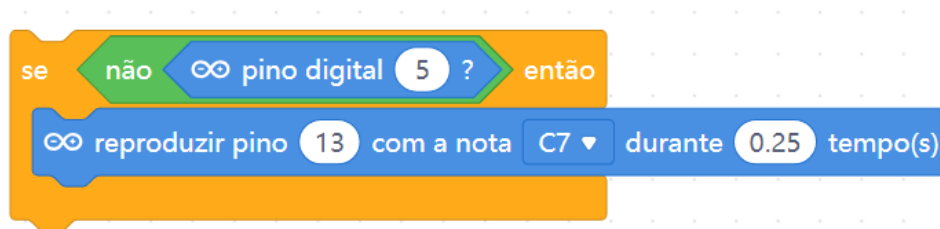
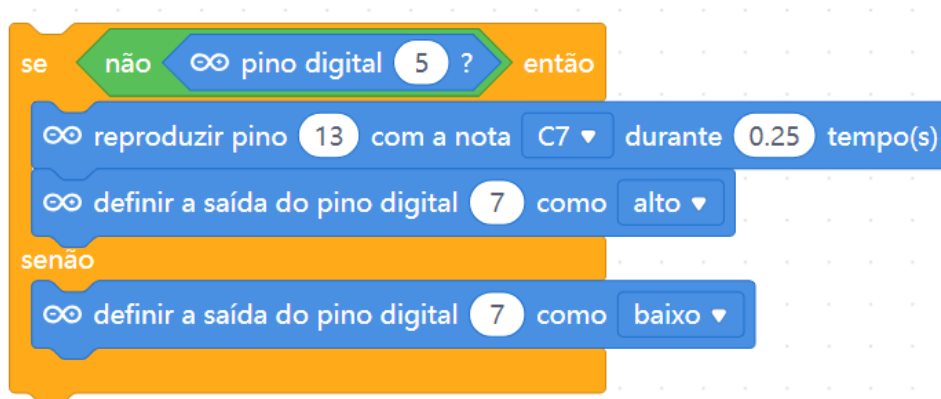
Nesta aula, os estudantes desenvolverão o projeto de Piano com o Arduino. Eles conhecerão dois componentes novos: o *buzzer*, para emitir sons, e o *push button*, que funciona como um botão.

1ª Parte: O professor deve iniciar a aula exibindo o projeto de piano, já implementado, para a turma. O projeto deve ser igual ao Projeto 1 da Folha de Atividades. Os estudantes podem brincar com o projeto enquanto o professor pergunta o que há de novidades neste projeto em relação aos projetos desenvolvidos anteriormente. Os estudantes devem apontar a funcionalidade e os componentes inéditos até então, que são o *buzzer* e o *push button*.

2ª Parte: O professor deve apresentar os novos componentes. Nesta etapa, os estudantes devem se dividir em grupos, de acordo com a disponibilidade dos kits, e manipular os componentes a medida em que são apresentados.

Ao apresentar o *buzzer*, o professor deve especificar a polaridade deste componente, onde o terminal positivo é maior do que o terminal negativo. O *buzzer* deve ter seu terminal positivo conectado a uma das portas digitais do Arduino. O professor também deve explanar brevemente sobre a capacidade deste componente em reproduzir músicas de modo similar ao que foi realizado com o mBot no 8º ano. Caso o professor julgue necessário, pode trazer algumas explicações sobre os conceitos de música, nota musical, frequência específica de cada nota.

Ao apresentar o *push button*, o professor deve especificar que este componente funciona como um botão. Ele possui quatro terminais e pode ser inserido no circuito de maneiras diferentes. No nosso projeto, ele funcionará como uma chave fechada enquanto não estiver pressionado, deixando a corrente passar. Ao ser pressionado, o *push button* interrompe a passagem da corrente. Ele deve ser conectado a uma das portas digitais do Arduino. Deste modo, ao ser pressionado ele enviará o valor *falso*, *LOW*, *baixo* ou *0* para o Arduino. Do modo como utilizaremos o *push button*, um resistor de 10k deve ser conectado ao mesmo terminal que é alimentado pela tensão de 5V. Nesta mesma trilha, conectamos o *push button* ao Arduino.

3ª Parte: O professor deve apresentar como a programação do projeto piano pode ser feita, mas sem mostrar o programa completo. Deve mostrar, por meio de dois exemplos, como verificar se o *push button* está ou não pressionado. É importante lembrar que o *push button* não vai conduzir corrente quando pressionado, por isso, vamos precisar do operador booleano *não*.

No primeiro exemplo, deve mostrar um bloco de código que verifica se o botão conectado ao pino digital 5 do Arduino foi pressionado. Caso a condição seja verdadeira, ou seja, caso não haja corrente no pino 5, então o *buzzer* tocará a nota Dó.

No segundo exemplo, o professor deve mostrar como definir ações para quando o *push button* não está pressionado. Para isso, será utilizado o comando *se...então...senão*. Neste exemplo, ao ser pressionado, o *push button* conectado ao pino digital 5 toca a nota dó e liga um LED conectado ao pino digital 7. Quando o *push button* não está pressionado, o LED é desligado.

4ª Parte: Os estudantes devem se dividir em grupos, de acordo com a quantidade de materiais disponíveis. Cada grupo deve seguir o Projeto 1 da Folha de Atividades. O professor deve oferecer suporte às equipes. Ao final, os estudantes devem brincar com o piano desenvolvido. Pode ser feita uma competição para ver quem consegue tocar uma música com o piano Arduino.

Caso alguma equipe finalize o projeto antes do previsto, o professor pode desafiá-la a implementar o Projeto 2 - Minipiano.

FOLHA DE ATIVIDADES

Continuando a nossa aventura pelo universo da Eletrônica, hoje é dia de fazer barulho com dois novos componentes: o *buzzer* e o *push button*!

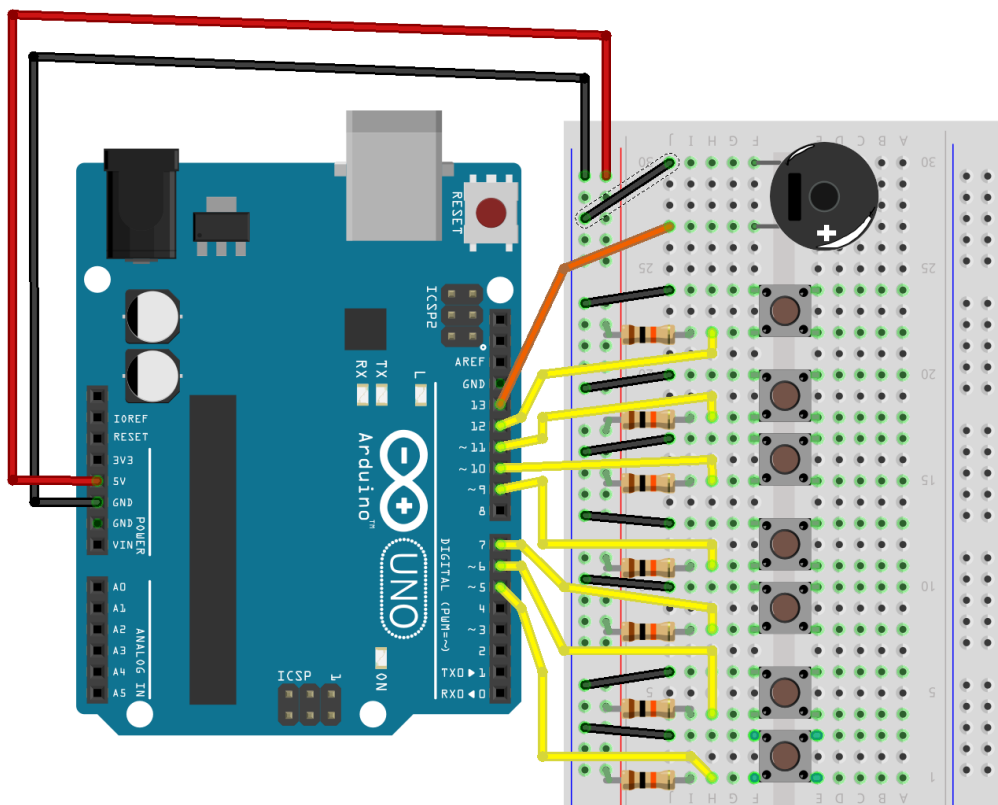
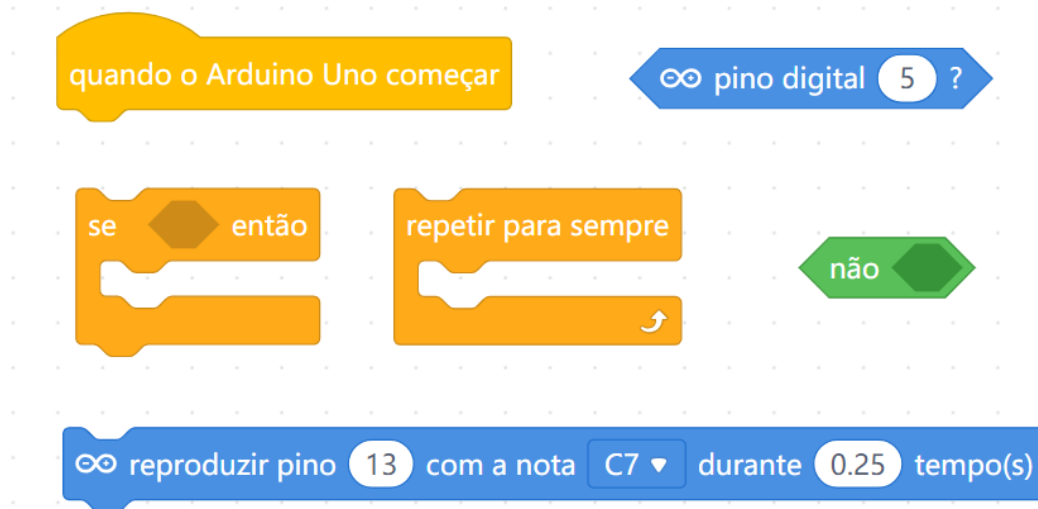
Projeto 1 – Piano Arduino

OBJETIVO: Construir um piano capaz de tocar as sete notas musicais: Dó, Ré, Mi, Fá, Sol, Lá, Si.

REQUISITOS: Este circuito terá sete *push buttons*, cada um correspondente a uma nota musical, e um *buzzer* que reproduzirá as notas quando os *push buttons* forem pressionados.

COMPONENTES NECESSÁRIOS:

Quantidade	Item
1	Arduino Uno
7	*Push button*
7	Resistor 10K
1	*Buzzer*
18	Jumpers do tipo macho/macho
1	Protoboard de 400 pontos

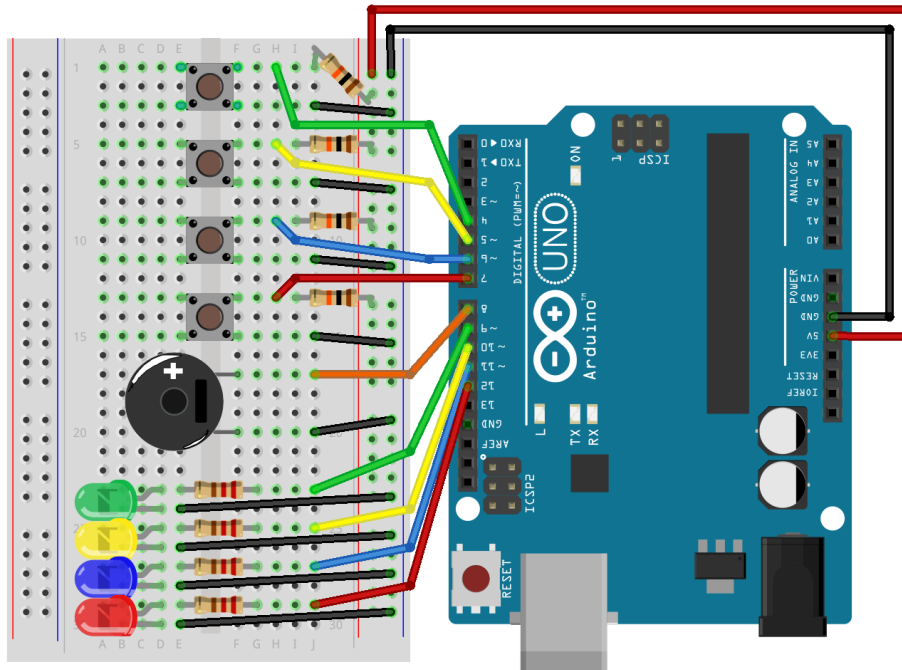
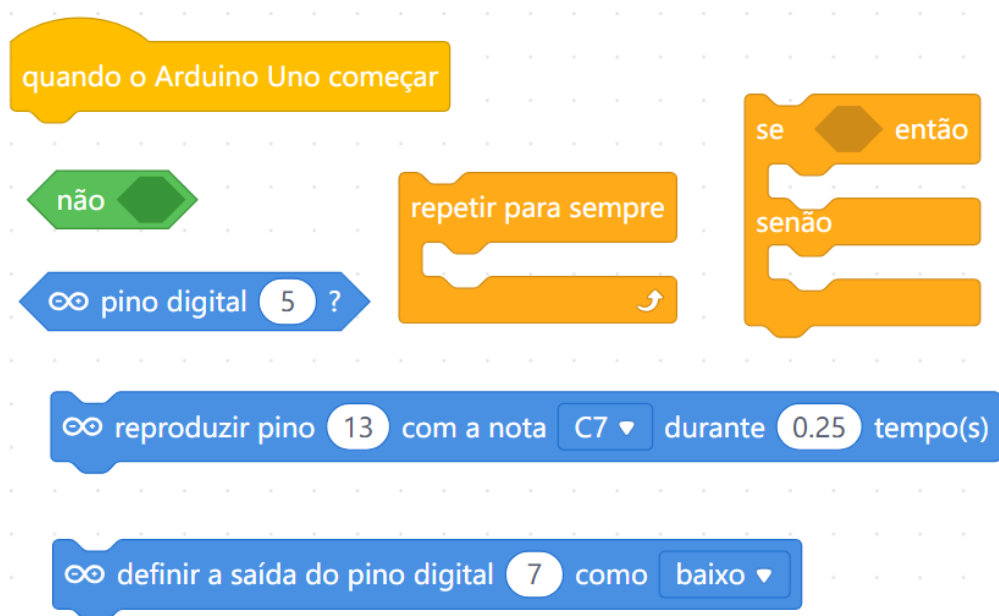
ESQUEMÁTICO:

PROGRAMAÇÃO NO MBLOCK: Para criar o programa do circuito piano, você deve utilizar os comandos abaixo, podendo usar outros blocos caso ache necessário. Como você é um programador expert, deve pensar num algoritmo que identifique quando um dos *push buttons* é pressionado e toque a nota correspondente a esta 'tecla' do piano no *buzzer*.

Projeto 2 – Minipiano Arduino

OBJETIVO: Construir um piano capaz de tocar quatro notas musicais (Dó, Ré, Mi, Fá) e acender um LED correspondente a cada nota, promovendo um efeito visual no circuito.

REQUISITOS: Este circuito terá quatro *push buttons*, cada um correspondente a uma nota musical. O *buzzer* deve reproduzir as notas musicais, de acordo com o *push button* pressionado. Cada 'tecla' do minipiano terá um LED associado, que deve ser aceso sempre que a tecla estiver pressionada.

COMPONENTES NECESSÁRIOS:

Quantidade	Item
1	Arduino Uno
4	*Push button*
4	Resistor 10K
4	LED (cores distintas)
4	Resistor 220R
1	*Buzzer*
20	Jumpers do tipo macho/macho
1	Protoboard de 400 pontos

ESQUEMÁTICO:

PROGRAMAÇÃO NO MBLOCK: Para criar o programa do circuito minipiano, você deve utilizar os comandos abaixo, podendo usar outros blocos caso ache necessário. Como você é um programador expert, deve pensar num algoritmo que identifique quando um dos *push buttons* é pressionado, toque a nota correspondente a esta 'tecla' do piano no *buzzer* e acenda o LED correspondente, e o apague quando a tecla for solta.

AULA 4 – Criando um Robô Dançarino

Nesta aula, os estudantes desenvolverão o projeto do Robô Dançarino, envolvendo servomotores e o Arduino.

Objetivos	Entender a funcionalidade do servomotor e ser capaz de incorporá-lo em um circuito; Compreender como ângulos podem ser usados para controlar a posição do braços de um servomotor; Criar um circuito que manipule diversos servomotores com comportamentos diferentes.
Conteúdo	Circuitos eletrônicos. Arduino. Servomotor (Micro Servo 9g SG90 TowerPro). Programação: sequências, loops e funções. Comandos do mBlock: Eventos, Controle, Pin e Meus Blocos.
Atividades	Projeto 1 – Robô dançarino.
Resultados de Aprendizagem	PC2. Usar passos básicos na solução de problemas algorítmicos para projetar soluções (por exemplo, declaração e exploração de problemas, exemplos de instâncias, design, implementação de uma solução, teste, avaliação). PC7. Descrever e analisar uma sequência de instruções que estão sendo seguidas (por exemplo, descrever o comportamento de um personagem em um videogame conforme orientado por regras e algoritmos). PCP10. Demonstrar compreensão dos algoritmos e sua aplicação prática. PCP11. Implementar soluções de problemas usando uma linguagem de programação, incluindo: comportamento de looping, instruções condicionais, lógica, expressões, variáveis e funções. PCP13. Identificar carreiras interdisciplinares que são aprimoradas pela ciência da computação. C7. Colaborar com colegas, especialistas e outras pessoas usando práticas colaborativas, como programação em pares, trabalho em equipes de projeto e participação em atividades de aprendizado ativo em grupo. C8. Apresentar as disposições necessárias para colaboração: fornecer feedback útil, integrar feedback, compreender e aceitar múltiplas perspectivas, socialização. CDC1. Reconhecer que os computadores modelam o comportamento inteligente (como encontrado em robótica, reconhecimento de fala e linguagem e animação por computador). CDC3. Reconhecer que os computadores são dispositivos que executam programas.

	CDC4. Identificar uma variedade de dispositivos eletrônicos que contêm processadores computacionais. CDC9. Descrever o que distingue os seres humanos das máquinas, concentrando-se na inteligência humana versus a inteligência das máquinas e nas maneiras pelas quais podemos comunicar. CDC10. Descrever maneiras pelas quais os computadores usam modelos de comportamento inteligente (por exemplo, movimento do robô, compreensão da fala e da linguagem, e visão computacional).
Materiais	Folha de atividades. Kit de eletrônica. mBlock. Materiais de papelaria e artesanato (Papelão, cola, EVA, fita adesiva, dentre outros).

DESCRIÇÃO DAS ATIVIDADES

Nesta aula, os estudantes desenvolverão o projeto de um Robô Dançarino. Eles irão conhecer o Micro Servo 9g e aplicá-lo no projeto para fazer movimentos da cabeça e dos braços do robô dançarino.

1ª Parte: O professor deve iniciar a aula exibindo o projeto robô dançarino com a sua música. Para isso, é necessário que o professor implemente previamente o projeto. O robô deve dançar de forma sincronizada com alguma música. O professor deve perguntar à turma o que há de novidade neste projeto em relação aos projetos que elaborados nas aulas anteriores.

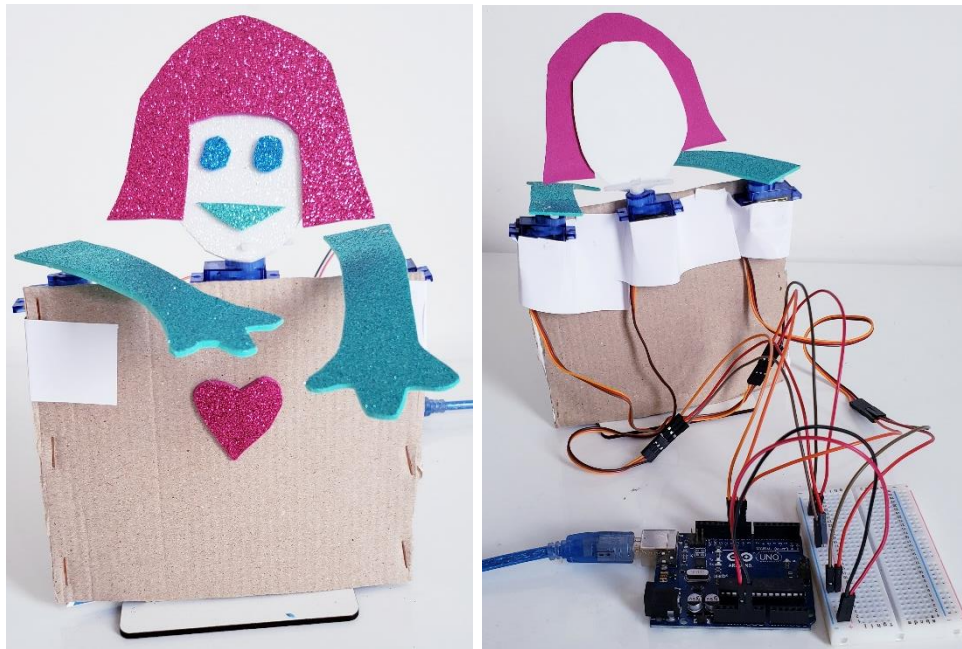
2ª Parte: O professor deve apresentar o Micro Servo 9g. Nesta etapa, os estudantes devem se dividir em grupos de acordo com a disponibilidade dos kits e manipular o servomotor à medida em que ele é apresentado.

O professor deve definir o conceito de servomotor como sendo uma máquina eletromecânica que recebe um sinal de controle e apresenta um movimento proporcional a este sinal. Quando um servomotor recebe um sinal, ele vai para a posição desejada, movendo-se apenas o necessário. A velocidade com que o motor se move também pode ser controlada.

O modelo de servomotor que trabalhamos é chamado Micro Servo 9g SG90 TowerPro, muito utilizado em projetos de robótica com Arduino. Este microsservo tem um grau de liberdade em seu eixo de 180º, ou seja, meia circunferência. Outra característica deste modelo é que ele não é muito preciso. Este servo possui 3 pinos: negativo (marrom), positivo/alimentação (vermelho) e controle (laranja). Ele também possui alguns “braços”, que podem ser parafusados.

3ª Parte: O professor deve demonstrar como a programação do servomotor pode ser feita. O comando específico para definir a posição do servo deve ser combinado com o comando espere.

O programa do robô dançarino estabelece sequências para os servomotores. As sequências sincronizadas oferecem a impressão de que o robô está “dançando”. O professor pode fazer alguns testes, mostrando os ângulos para fazer com que o robô abra ou feche os braços.

4ª Parte: Os estudantes devem se dividir em grupos, de acordo com a quantidade de materiais disponíveis. Cada grupo deve confeccionar a base do robô dançarino com materiais de papelaria disponíveis. As imagens a seguir oferecem exemplos, mas o professor pode definir a estrutura que desejar, de acordo com os materiais que puder reunir.

5ª Parte: Cada grupo deve seguir o Projeto 1 da Folha de Atividades. O professor deve acoplar os braços aos servos antes da aula, de modo que todos os servos estejam posicionados em 0º, caso contrário, os estudantes poderão ter dificuldades de posicionar o braço corretamente. Os estudantes devem montar o circuito seguindo o esquemático e, depois, acoplar os motores aos braços e à cabeça do robô. É interessante que cada grupo escolha previamente uma música para combinar com os movimentos do robô. O professor deve oferecer suporte aos grupos durante a etapa de desenvolvimento.

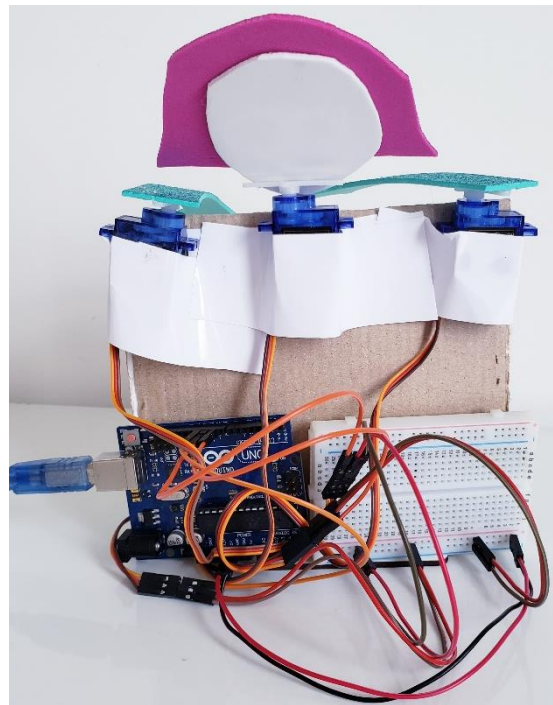
6ª Parte: Ao final da aula, cada grupo deve apresentar seus projetos.

FOLHA DE ATIVIDADES

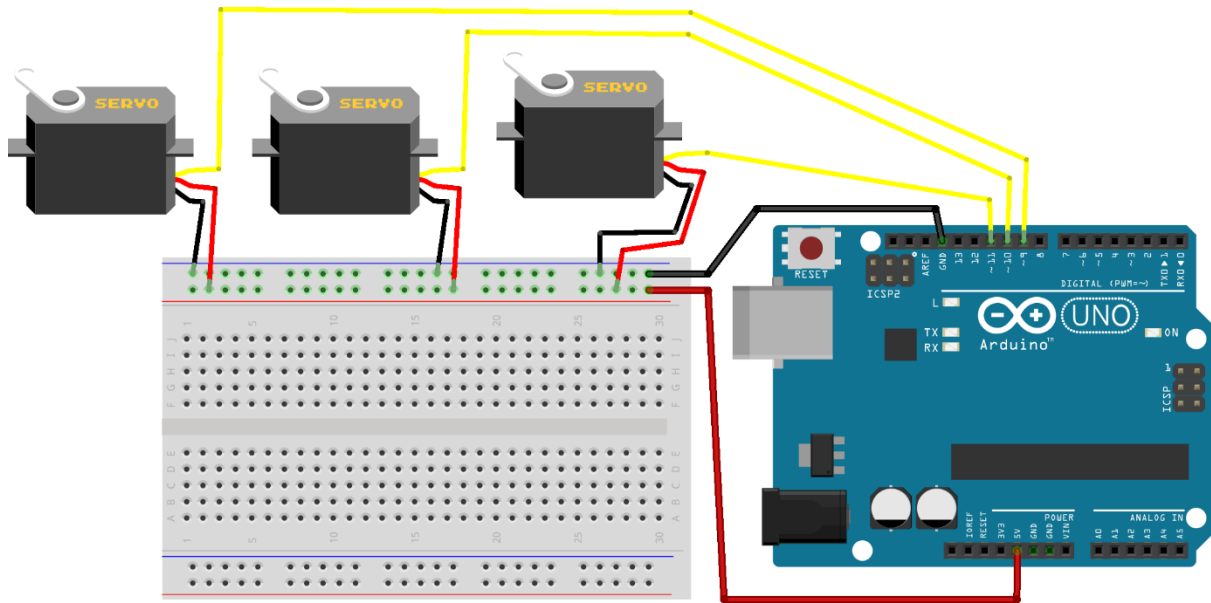
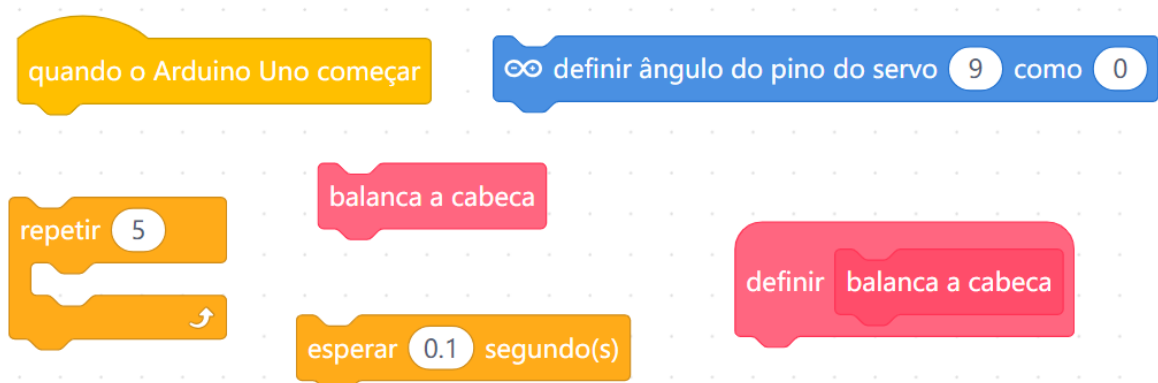
Continuando a nossa aventura pelo universo da eletrônica, hoje é dia de dançar com um novo componente: o Micro Servo 9g SG90 TowerPro! Este servomotor é capaz de posicionar seu "braço" entre 0º e 180º.

Projeto 1 – Robô Dançarino

OBJETIVO: Construir um robô que execute movimentos dos braços e da cabeça de maneira sincronizada com uma música de escolha do projetista.

REQUISITOS: Este circuito terá três servomotores. O projetista deve escolher uma música e criar uma sequência para cada servomotor, simulando a coreografia da música escolhida. Antes da montagem do circuito, o projetista deve criar uma base para o robô, incluindo os braços e a cabeça. Um servo faz o movimento da cabeça, um servo faz o movimento do braço direito e outro servo faz o movimento do braço esquerdo.

COMPONENTES NECESSÁRIOS:

Quantidade	Item
1	Arduino Uno
3	Micro Servo 9g SG90 TowerPro
1	Protoboard de 400 pontos
11	Jumpers do tipo macho/macho

ESQUEMÁTICO:

PROGRAMAÇÃO NO MBLOCK: Para criar o programa do Robô Dançarino, você deve utilizar os comandos abaixo, podendo usar outros blocos caso ache necessário. Como você é um programador expert, deve pensar num algoritmo que estabeleça uma sequência de ações para os servos. Para que seu programa não fique muito grande, você pode utilizar a criação de blocos, na categoria Meus Blocos. Por exemplo, definir um bloco para quando o robô balança a cabeça.

O comando que especifica a posição do servomotor é *Definir Ângulo Do Pino Servo...* Nele, você deve especificar em que porta digital do Arduino o Servo está conectado e o ângulo em que seu braço deve ser posicionado. O modelo de servomotor que trabalhamos só se posiciona entra 0º e 180º.

AULA 5 – Criando um Robô Guardião do Tesouro

Nesta aula, os estudantes irão trabalhar com o sensor ultrassônico e irão alterar o projeto do Robô Dançarino, transformando-o no Robô Guardião do Tesouro.

Objetivos	Entender a funcionalidade do sensor ultrassônico e ser capaz de incorporá-lo em um circuito. Alterar circuitos já montados a partir da análise do esquemático. Criar um circuito que manipule diversos servomotores com comportamentos diferentes.
Conteúdo	Circuitos eletrônicos. Arduino. Sensor ultrassônico (HSR04). Programação: sequências, loops e condições. Comandos do mBlock: Eventos, Controle, Pin e Sensores.
Atividades	Projeto 1 – Robô Guardião do Tesouro.
Resultados de Aprendizagem	PC2. Usar passos básicos na solução de problemas algorítmicos para projetar soluções (por exemplo, declaração e exploração de problemas, exemplos de instâncias, design, implementação de uma solução, teste, avaliação). PC7. Descrever e analisar uma sequência de instruções que estão sendo seguidas (por exemplo, descrever o comportamento de um personagem em um videogame conforme orientado por regras e algoritmos). PCP10. Demonstrar compreensão dos algoritmos e sua aplicação prática. PCP11. Implementar soluções de problemas usando uma linguagem de programação, incluindo: comportamento de looping, instruções condicionais, lógica, expressões, variáveis e funções. PCP13. Identificar carreiras interdisciplinares que são aprimoradas pela ciência da computação. C7. Colaborar com colegas, especialistas e outras pessoas usando práticas colaborativas, como programação em pares, trabalho em equipes de projeto e participação em atividades de aprendizado ativo em grupo. C8. Apresentar as disposições necessárias para colaboração: fornecer feedback útil, integrar feedback, compreender e aceitar múltiplas perspectivas, socialização. CDC1. Reconhecer que os computadores modelam o comportamento inteligente (como encontrado em robótica, reconhecimento de fala e linguagem e animação por

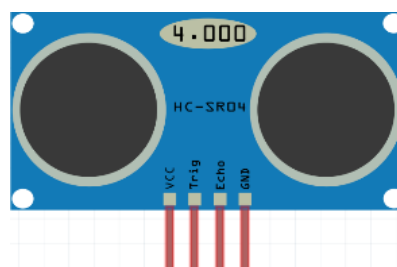
	computador). CDC3. Reconhecer que os computadores são dispositivos que executam programas. CDC4. Identificar uma variedade de dispositivos eletrônicos que contêm processadores computacionais. CDC9. Descrever o que distingue os seres humanos das máquinas, concentrando-se na inteligência humana versus a inteligência das máquinas e nas maneiras pelas quais podemos comunicar. CDC10. Descrever maneiras pelas quais os computadores usam modelos de comportamento inteligente (por exemplo, movimento do robô, compreensão da fala e da linguagem, e visão computacional).
Materiais	Folha de atividades. Kit de eletrônica. mBlock.

DESCRIÇÃO DAS ATIVIDADES

Nesta aula, os estudantes irão trabalhar com o sensor ultrassônico. Os estudantes devem alterar o projeto do Robô Dançarino, transformando-o no Robô Guardião do Tesouro. Ao detectar a presença de algo próximo, o robô fecha os braços, protegendo o tesouro. No exemplo a seguir, o tesouro é o anão Atchim!

1ª Parte: O professor deve iniciar a aula exibindo o projeto robô guardião do tesouro. Para isso, é necessário que o professor implemente previamente o projeto. O robô deve fechar os braços sempre que o sensor ultrassônico detecta um obstáculo a uma distância menor do que 20 cm. O professor deve perguntar à turma o que há de novidade neste projeto em relação aos projetos que foram elaborados nas aulas anteriores.

2ª Parte: O professor deve apresentar o sensor Ultrassônico HSR04. Nesta etapa, os estudantes devem se dividir em grupos de acordo com a disponibilidade dos kits e manipular o sensor à medida em que ele é apresentado.

O professor deve definir o conceito de sensor ultrassônico como um sensor que utiliza módulos transmissores e receptores ultrassônicos para determinar a distância a um objeto, de modo similar ao que os morcegos fazem. O sensor ultrassônico HSR04 é capaz de medir distâncias entre 3 e 400 cm. Pode ser usado nos projetos para ajudar robôs a evitar obstáculos ou em outros projetos relevantes de medição de distância. Este sensor é similar ao módulo ultrassônico do mBot que foi trabalhado no livro do 8º ano.

Este sensor possui quatro terminais, *VCC*, *Trig*, *Echo* e *GND*. O transmissor (*Trig*) envia um sinal que é um som com alta frequência. Quando o sinal encontra um objeto, ele é refletido e o receptor (*Echo*) o recebe. O cálculo da distância é feito de acordo com o tempo em que o sinal emitido demorou para ser recebido.

3ª Parte: O professor deve demonstrar como a medição de distâncias pode ser feita através do sensor ultrassônico. O comando específico da categoria Sensores pode ser utilizado passando como parâmetro o número dos pinos digitais onde os terminais *Trig* e *Echo* estão conectados ao Arduino. O bloco sensor azul a seguir retorna o valor da distância, em centímetros, entre o sensor ultrassônico e o objeto mais próximo dentro do seu alcance.

ler pin trigonométrico do sensor ultrasónico 13 pin de eco 12 < 20

O algoritmo para este projeto é o seguinte:

1. O robô inicia com os braços abertos;
2. Se identificar um objeto a uma distância menor do que 20 cm, então o robô fecha os braços. O projetista também pode definir um comportamento para a cabeça do robô.

4ª Parte: Os estudantes devem se dividir em grupos, de acordo com a quantidade de materiais disponíveis. Cada grupo deve alterar o projeto do robô dançarino, transformando-o em um robô guardião do tesouro.

5ª Parte: Cada grupo deve seguir o Projeto 1 da Folha de Atividades. Os estudantes devem alterar o circuito seguindo o esquemático e, depois, programá-lo. O professor deve oferecer suporte aos grupos durante a etapa de desenvolvimento.

6ª Parte: Ao final da aula, cada grupo deve apresentar seus projetos. Pode ser feita uma competição com os robôs para ver quem consegue roubar o tesouro sem ser identificado pelo robô.

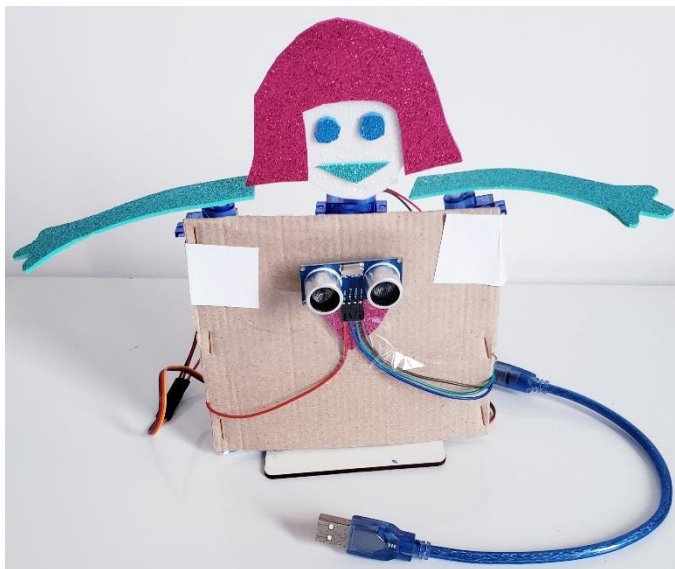
FOLHA DE ATIVIDADES

Continuando a nossa aventura pelo universo da eletrônica, hoje é dia de conhecer um novo componente: o Sensor Ultrassônico HSR04! Este sensor é capaz de medir distâncias entre 3 centímetros e 4 metros.

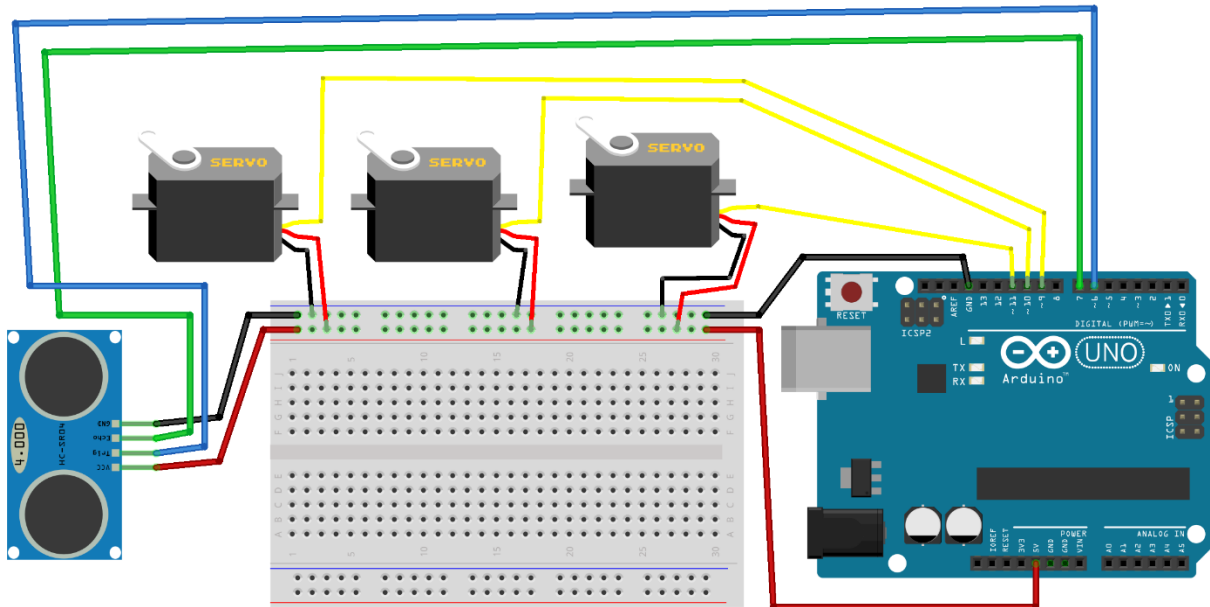
Projeto 1 – Robô Guardião do Tesouro

OBJETIVO: Construir um robô que identifique intrusos e proteja um objeto.

REQUISITOS: Este circuito terá três servomotores e um sensor ultrassônico. Um servo faz o movimento da cabeça, um servo faz o movimento do braço direito e o outro servo faz o movimento do braço esquerdo. O projetista deve definir um comportamento para os braços e a cabeça do robô se um objeto for detectado pelo sensor ultrassônico a uma distância menor do que 20 cm. O projetista pode alterar este limite de 20 cm caso julgue necessário durante a fase de teste do projeto.

O projetista deve aproveitar o projeto criado de Robô Dançarino, e promover alterações neste circuito, adicionando o sensor ultrassônico.

COMPONENTES NECESSÁRIOS:

Quantidade	Item
1	Arduino Uno
1	Sensor Ultrassônico HSR04
3	Micro Servo 9g SG90 TowerPro
1	Protoboard de 400 pontos
15	Jumpers do tipo macho/macho

ESQUEMÁTICO:

PROGRAMAÇÃO NO MBLOCK: Para criar o programa do Robô Guardião do Tesouro, você deve utilizar os comandos abaixo, podendo usar outros blocos caso ache necessário. Como você é um programador expert, deve pensar num algoritmo onde o robô fique com os braços abertos e, se identificar um objeto a uma distância menor do que 20 cm, então o robô fecha os braços. O projetista também pode definir um comportamento para a cabeça do robô.

O comando que especifica a posição do servomotor é *Ler pin trigonométrico do sensor ultrassônico...* Nele, você deve especificar em que porta digital do Arduino o terminal *Trig* do sensor está conectado e o pino digital onde o terminal *Echo* do sensor está conectado.

ler pin trigonométrico do sensor ultrasónico 13 pin de eco 12

AULA 6 – Como funciona o Computador?

Nesta aula, os estudantes irão trabalhar com a noção de arquitetura de hardware dos computadores.

Objetivos	Compreender que há um padrão na arquitetura dos computadores modernos. Classificar os componentes do computador de acordo com sua função na arquitetura de hardware. Entender como a arquitetura de hardware e software de um computador são essenciais para o seu funcionamento.
Conteúdo	Arquitetura de hardware: Processador; Memória; Dispositivos de entrada e saída. Arquitetura de software: Instrução; Programa ou software; Sistema operacional.
Atividades	Leitura em sala em conjunto com apresentação. Classificando os componentes do computador.
Resultados de Aprendizagem	PC14. Compreender a noção de hierarquia e abstração na computação, incluindo linguagens de alto nível, tradução, conjunto de instruções, e circuitos lógicos. CDC3. Reconhecer que os computadores são dispositivos que executam programas. CDC5. Demonstrar uma compreensão da relação entre hardware e software.
Materiais	Folha de atividades. Kit de eletrônica. mBlock.

DESCRIÇÃO DAS ATIVIDADES

Nesta aula, os estudantes irão trabalhar com a noção de arquitetura de hardware dos computadores. Eles já tiveram uma experiência com este assunto no livro do sexto ano. Desta vez, entretanto, este assunto será abordado com um tom mais teórico, condizente com o nível de entendimento dos estudantes.

1ª Parte: O professor deve iniciar a aula perguntando quem lembra o que é dispositivo de entrada e saída de dados, memória e processador. O professor pode citar a aula deste tema no material do sexto ano. Os alunos podem então iniciar a leitura disponível em Tópicos Relevantes. É importante que a leitura seja acompanhada de explicações do professor apoiadas em um computador de verdade. O ideal é que o computador esteja desmontado ou com a tampa lateral do gabinete aberta.

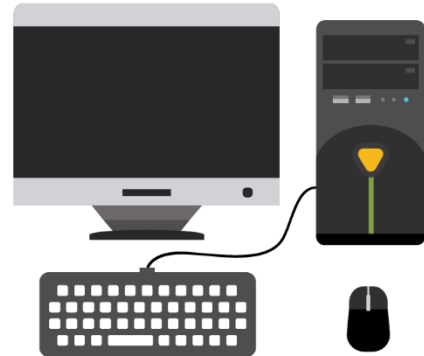
À medida em que o professor apresenta a arquitetura de hardware do computador, também deve pontuar características da arquitetura de software, como os conceitos de instrução, programa e sistema operacional:

- **Instrução**: Descrição de uma ação a ser realizada pelo processador e que fica armazenada na memória.
- **Programa** ou **software**: Conjunto de instruções armazenado na memória para o processador executar. Convém explicar o nível de abstração das linguagens de programação de alto e baixo nível.
- **Sistema operacional**: programa que é executado a todo tempo e que determina quais programas vão executar, quando, e quais recursos eles irão utilizar. O sistema operacional é responsável por fazer a interface entre o software e o hardware.

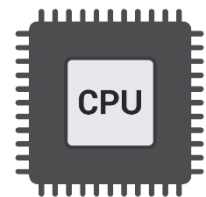
2ª Parte: Os estudantes devem realizar a atividade prevista na Folha de Atividades. O professor deve reservar um tempo da aula para que os estudantes façam a atividade e, ao final desse intervalo, fazer a correção em grupo.

TÓPICOS RELEVANTES

Um computador é uma máquina destinada ao processamento de informação. Os computadores costumam ser classificados de acordo com a capacidade de processamento e sua função. Assim, são alguns tipos de computadores: microcomputadores, videogames, servidores, supercomputadores, desktops, smartphones, sistemas embarcados (lembra do Arduino?), dentre outros.

Na prática, os computadores são dispositivos eletrônicos programáveis que recebem dados, executam operações lógicas e matemáticas em alta velocidade e exibem o resultado destas operações.

Desde os primeiros computadores da década de 1940, a tecnologia utilizada nos computadores digitais mudou bastante. Apesar disso, quase todos os computadores ainda utilizam uma variante da arquitetura proposta pelo matemático John von Neumann na primeira metade do século XX. A chamada Arquitetura de von Neumann tem quatro partes principais: unidade lógica e aritmética, unidade de controle, memória e dispositivos de entrada e saída. A unidade lógica e aritmética e a unidade de controle se combinam formando o processador. Para interligar todas essas partes, existe a placa-mãe, que conecta os dispositivos através de barramentos e fios.

Processador

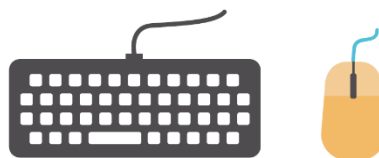
Todo computador possui um processador, também conhecido como CPU (*Central Processing Unit).* O processador é composto pela unidade lógica e aritmética, unidade de controle e registradores:

- **Unidade Lógica e Aritmética (ULA):** é a parte do processador que executa as operações aritméticas e lógicas entre números. As ULAs podem ter apenas as operações de adição e subtração, mas algumas ULAs podem incluir outras operações como a multiplicação e divisão. Também possuem as operações lógicas (por exemplo: E, Ou, Não).
- **Unidade de controle (UC):** é a parte do processador que controla seu funcionamento interno além de gerar sinais que controlam o funcionamento externo à CPU. A função mais importante da UC é executar as instruções dos programas. Ela recebe uma instrução da memória, interpreta-a e gera os sinais para as outras partes do computador (ULA, memória, dispositivos de entrada e saída) para executá-la.
- **Registradores:** são a parte do processador dedicada ao armazenamento de dados. Os registradores são necessários pois evitam que o processador acesse a memória do computador continuamente. Cada registrador é uma pequena porção de memória que pode ser lida e escrita de maneira mais rápida que a memória primária do computador.

Memória

Todo computador precisa armazenar dados e a parte responsável por isso é a memória. A memória pode ser de leitura e escrita ou apenas de leitura. Leitura significa o processador receber dados da memória, enquanto escrita significa o processador enviar dados para a memória.
Um computador normalmente tem mais de um componente de memória. Geralmente, a memória está dividida em memória primária e memória secundária:

- **Memória primária:** é uma memória que pode ser acessada diretamente pelo processador. Fazem parte desta memória:
    - **Memória cache:** memória de acesso rápido, situada muito próxima à ULA do processador.
    - **Memória de acesso aleatório (Random Access Memory - RAM):** é uma memória composta por células de armazenamento numeradas, capazes de armazenar instruções a serem executadas pelo processador, ou dados necessários para que o processador execute alguma instrução. É uma memória volátil, ou seja, quando o computador é desligado os dados são perdidos.
    - **Memória apenas de leitura (Read-Only Memory - ROM):** é uma memória não volátil e que só pode ser lida. Esse tipo de memória armazena o *firmware* do computador, que é um programa que controla as funções principais do hardware de modo que o computador possa ser ligado e estabeleça comunicação adequada com os dispositivos de entrada e saída.
- **Memória Secundária:** é usada para gravar os dados em geral, como fotos e documentos. Este tipo de memória não perde os dados com o desligamento do computador. A principal memória secundária do computador é o **Disco Rigído (Hard Disk - HD)**. Ela também armazena os programas instalados de forma que possam ser executados sempre que o computador for ligado. Além disso, armazena os arquivos com dados de modo que possam sempre ser acessados quando o computador está ligado. Alguns computadores mais atuais usam a **memória SSD (solid-state drive)** no lugar do HD. As memórias secundárias possuem acesso mais lento que as memórias primárias. Também são considerados memória secundária de um computador os CDs, DVDs, e pen drives.

Dispositivos de entrada e saída

Os dispositivos de entrada e saída são também chamados de periféricos. Dispositivos de entrada codificam a informação passada pelo usuário do computador em tipos de dados digitais que podem ser processados pelo computador. São exemplos de dispositivos de entrada: teclado, *mouse*, *webcam*, scanner, microfone e *touchpad*.

Os dispositivos de saída decodificam a informação de modo que ela possa ser entendida pelo usuário do computador. São exemplos de dispositivos de saída: monitor, projetor, alto-falantes, fones de ouvido e impressora.

Alguns dispositivos periféricos podem ser tanto de entrada como de saída. Por exemplo, placas de rede com fio ou sem fio, dispositivos *Bluetooth* são dispositivos de entrada e saída.

Outros elementos

Além dos elementos processador, memória e dispositivos de entrada e saída, existem outros componentes que fazem parte da arquitetura do computador. São eles: a placa-mãe, a fonte de alimentação e o gabinete que armazena todos os componentes.

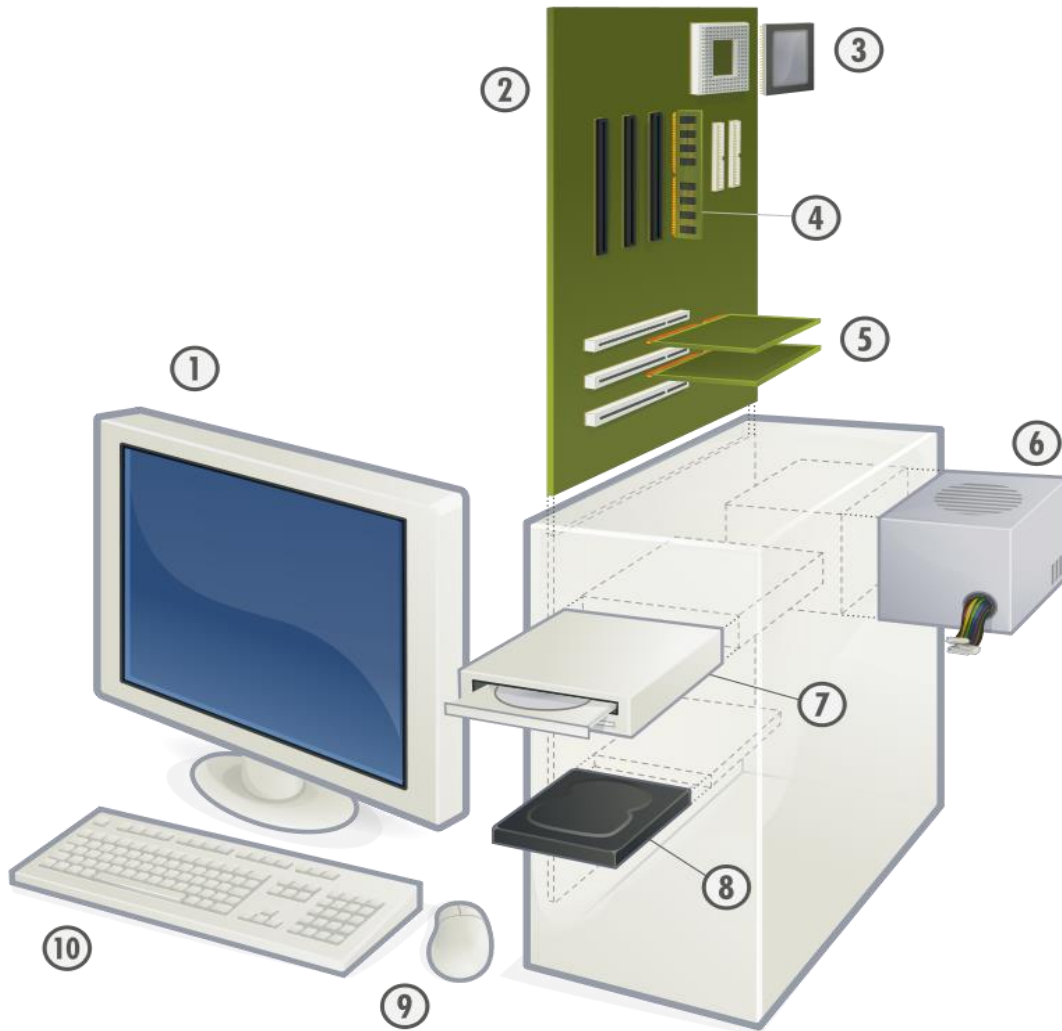
A placa-mãe é uma placa de circuito com diversas entradas, chamas *slots*, que conectam todos os componentes do computador. Através da placa-mãe processador, memória RAM, disco rígido, placa gráfica e os demais componentes são interligados. A organização de uma placa-mãe varia conforme o modelo e fabricante, mas a função principal sempre se mantém. A principal função da placa-mãe é permitir o tráfego de informação, mas ela também alimenta alguns periféricos com a energia elétrica que recebe da fonte de alimentação.

A fonte de alimentação do computador é um componente que converte a voltagem de energia elétrica disponível nas tomadas, algo em torno de 110 ou 220 volts, para voltagens menores que são suportadas pelos componentes do computador, como 12 volts ou menos. Em computadores como notebooks, a fonte também costuma ser uma bateria, permitindo que o computador funcione desconectado da internet.

O gabinete computador é o compartimento que tem a maioria dos componentes, incluindo a placa-mãe e os dispositivos a ela acoplados. Normalmente, o monitor, o teclado e o mouse ficam de fora do gabinete. Muitas vezes o gabinete é chamado de CPU, mas sabemos que CPU é, na verdade, o processador do computador.

FOLHA DE ATIVIDADES

Exercício 1 – Partes do Computador

A Figura abaixo exibe um computador desktop tradicional com suas partes evidenciadas. Coloque o nome do dispositivo e classifique, quando pertinente, a parte listada como sendo processador, memória ou dispositivo de entrada e saída:

01- ______________________________ - ____

02- ______________________________ - ____

03- ______________________________ - ____

04- ______________________________ - ____

05- ______________________________ - ____

06- ______________________________ - ____

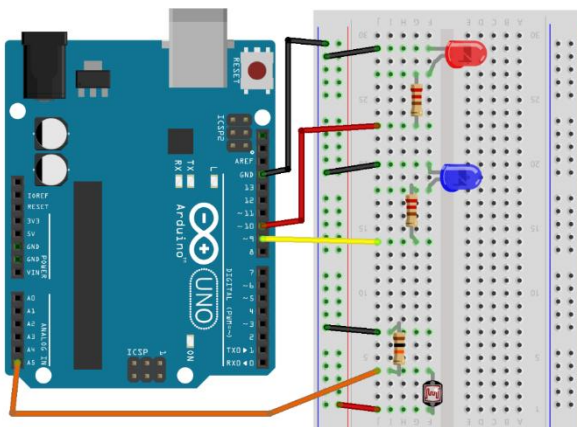
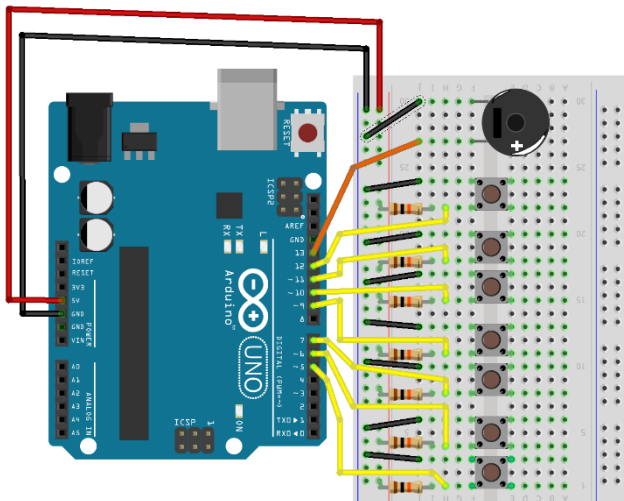
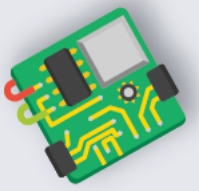
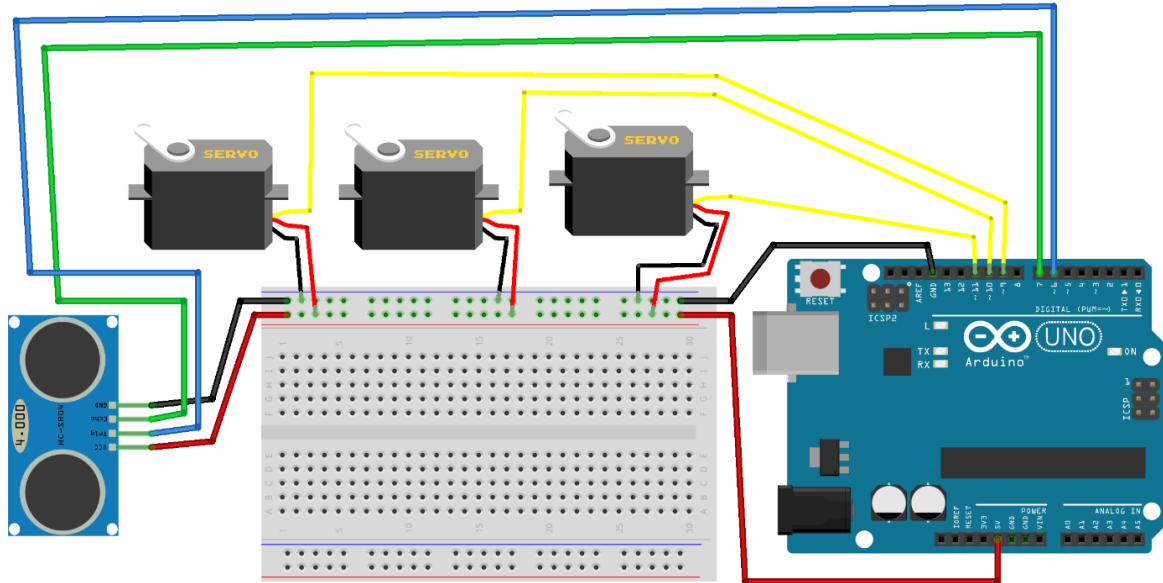
07- ______________________________ - ____

08- ______________________________ - ____

09- ______________________________ - ____

Exercício 2 - Processamento, entrada ou saída?

Durante esta unidade criamos vários projetos com Arduino. Muitos destes projetos recebem informações do ambiente ou do usuário, processam estas informações e retornam alguma resposta. Para cada um dos projetos a seguir, identifique os componentes que estão encarregados do processamento e memória, entrada de dados ou saída de dados.

Medindo a luminosidade do ambiente:

Piano com Arduino:

Robô Guardião do Tesouro:

SENSOR ULTRASSÔNICO:	SERVO MOTOR:	ARDUINO:

AULA 7 – Criando seu Primeiro Protótipo com o Arduino

Nesta aula, os estudantes devem criar um protótipo de um produto de seu interesse através de um circuito com o Arduino e outros componentes eletrônicos aprendidos ao longo da unidade.

Objetivos	Aplicar o conhecimento sobre Arduino, sensores, motores, buzzer e LEDs para projetar circuitos com comportamento autônomo.
Conteúdo	Circuitos eletrônicos.
	Arduino.
	Componentes: LEDs, Buzzer, Push buttons, LDR, Sensor ultrassônico (HSR04).
	Programação: sequências, loops, condições e funções.
	Comandos do mBlock: Eventos, Controle, Pin, Operadores, Variáveis, Sensores e Meus Blocos.
Atividades	Projeto Final – Circuito com Arduino.
Resultados de Aprendizagem	PC2. Usar passos básicos na solução de problemas algorítmicos para projetar soluções (por exemplo, declaração e exploração de problemas, exemplos de instâncias, design, implementação de uma solução, teste, avaliação).
	PC7. Descrever e analisar uma sequência de instruções que estão sendo seguidas (por exemplo, descrever o comportamento de um personagem em um videogame conforme orientado por regras e algoritmos).
	PCP10. Demonstrar compreensão dos algoritmos e sua aplicação prática.
	PCP11. Implementar soluções de problemas usando uma linguagem de programação, incluindo: comportamento de looping, instruções condicionais, lógica, expressões, variáveis e funções.
	PCP13. Identificar carreiras interdisciplinares que são aprimoradas pela ciência da computação.
	C7. Colaborar com colegas, especialistas e outras pessoas usando práticas colaborativas, como programação em pares, trabalho em equipes de projeto e participação em atividades de aprendizado ativo em grupo.
	C8. Apresentar as disposições necessárias para colaboração: fornecer feedback útil, integrar feedback, compreender e aceitar múltiplas perspectivas, socialização.
	CDC1. Reconhecer que os computadores modelam o comportamento inteligente (como encontrado em robótica, reconhecimento de fala e linguagem e animação por

	computador).
	CDC3. Reconhecer que os computadores são dispositivos que executam programas.
	CDC4. Identificar uma variedade de dispositivos eletrônicos que contêm processadores computacionais.
	CDC9. Descrever o que distingue os seres humanos das máquinas, concentrando-se na inteligência humana versus a inteligência das máquinas e nas maneiras pelas quais podemos comunicar.
	CDC10. Descrever maneiras pelas quais os computadores usam modelos de comportamento inteligente (por exemplo, movimento do robô, compreensão da fala e da linguagem, e visão computacional).
Materiais	Folha de atividades.
	Kit de eletrônica.
	mBlock.

DESCRIÇÃO DAS ATIVIDADES

Nesta aula, os estudantes devem criar um circuito com o Arduino e os demais componentes eletrônicos aprendidos ao longo da unidade.

1ª Parte: O professor deve iniciar a aula listando com os estudantes tudo o que eles aprenderam até então relacionado a eletrônica, robótica e Arduino. É importante que esta lista seja escrita no quadro, por exemplo. Devem ser listadas as possibilidades de circuitos com o Arduino e os componentes trabalhados, as limitações dos circuitos com Arduino, as estratégias utilizadas para programar comportamento autônomo com os sensores, etc.

2ª Parte: O professor apresenta o trabalho final aos estudantes. Eles serão livres para implementar um circuito com o Arduino que tenha diversas funcionalidades trabalhadas ao longo da unidade. Os estudantes devem se guiar pela Folha de Atividades, que especifica algumas funcionalidades que o projeto deve ter. Todo o tempo da aula deve ser reservado para que os grupos trabalhem em seu protótipo.

FOLHA DE ATIVIDADES

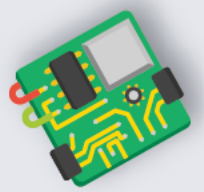
Nesta unidade, já trabalhamos com diversos tipos de circuitos com Arduino. Também aprendemos diversos aspectos relevantes de robótica e eletrônica, como as limitações inatas dos componentes eletrônicos e as dificuldades de programar um comportamento autônomo e inteligente. Dentre várias possibilidades:

- Aprendemos a ligar e desligar LEDs;
- Aprendemos a fazer barulho com o *Buzzer*;
- Aprendemos a programar circuitos com botões;
- Aprendemos a criar movimentos com o servomotor;
- Aprendemos a medir luminosidade com o sensor LDR;
- Aprendemos a medir distâncias com o sensor ultrassônico.

Protótipo com Arduino

Agora que você conhece muito sobre eletrônica com Arduino, crie seu primeiro protótipo de um produto eletrônico. Seu circuito deve ser o mais inteligente possível e pode se parecer com um robô. Você também pode usar a sua criatividade e embarcá-lo em criações artesanais, como nos projetos do robô dançarino e do robô guardião do tesouro.

Seu circuito deve ter pelo menos quatro dos componentes a seguir, sendo que um deles deve ser obrigatoriamente um sensor:

- LED;
- *Buzzer*;
- *Push button*;
- Servomotor;
- Sensor de luminosidade;
- Sensor Ultrassônico.

AULA 8 – Apresentando seu Primeiro Protótipo com o Arduino

Nesta aula, os estudantes devem apresentar o protótipo de produto criado com o Arduino para a comunidade.

Objetivos	Apresentar projetos de maneira formal, listando as funcionalidades empregadas.
Conteúdo	Circuitos eletrônicos.
	Arduino.
	Componentes: LEDs, Buzzer, Push buttons, LDR, Sensor ultrassônico (HSR04).
	Programação: sequências, loops, condições e funções.
	Comandos do mBlock: Eventos, Controle, Pin, Operadores, Variáveis, Sensores e Meus Blocos.
	Apresentação de Slides.
Atividades	Apresentando o protótipo.
Resultados de Aprendizagem	PC2. Usar passos básicos na solução de problemas algorítmicos para projetar soluções (por exemplo, declaração e exploração de problemas, exemplos de instâncias, design, implementação de uma solução, teste, avaliação).
	PC7. Descrever e analisar uma sequência de instruções que estão sendo seguidas (por exemplo, descrever o comportamento de um personagem em um videogame conforme orientado por regras e algoritmos).
	PCP10. Demonstrar compreensão dos algoritmos e sua aplicação prática.
	PCP11. Implementar soluções de problemas usando uma linguagem de programação, incluindo: comportamento de looping, instruções condicionais, lógica, expressões, variáveis e funções.
	PCP13. Identificar carreiras interdisciplinares que são aprimoradas pela ciência da computação.
	C7. Colaborar com colegas, especialistas e outras pessoas usando práticas colaborativas, como programação em pares, trabalho em equipes de projeto e participação em atividades de aprendizado ativo em grupo.
	C8. Apresentar as disposições necessárias para colaboração: fornecer feedback útil, integrar feedback, compreender e aceitar múltiplas perspectivas, socialização.

	CDC1. Reconhecer que os computadores modelam o comportamento inteligente (como encontrado em robótica, reconhecimento de fala e linguagem e animação por computador). CDC3. Reconhecer que os computadores são dispositivos que executam programas. CDC4. Identificar uma variedade de dispositivos eletrônicos que contêm processadores computacionais. CDC9. Descrever o que distingue os seres humanos das máquinas, concentrando-se na inteligência humana versus a inteligência das máquinas e nas maneiras pelas quais podemos comunicar. CDC10. Descrever maneiras pelas quais os computadores usam modelos de comportamento inteligente (por exemplo, movimento do robô, compreensão da fala e da linguagem, e visão computacional).
Materiais	Software para apresentação de slides (PowerPoint ou LibreOffice Impress) ou Google Apresentações. Kit de eletrônica. mBlock.

DESCRIÇÃO DAS ATIVIDADES

Nesta aula, os estudantes devem criar uma apresentação de slides, descrevendo seu protótipo com o Arduino. Em um segundo momento, os estudantes devem apresentar o projeto criado, fazendo a apresentação de slides e demonstrando o uso do circuito.

1ª Parte: O professor deve iniciar a aula falando que os estudantes devem elaborar uma apresentação de slides que evidencie as principais características do protótipo criado na aula anterior. A apresentação pode ser criada no software específico (PowerPoint ou LibreOffice Impress) ou através do Google Apresentações. O professor deve reservar um tempo da aula para que os estudantes elaborem a apresentação, que deve conter:

- Slide de capa, com título do projeto e nome dos integrantes do grupo;
- Objetivo do produto desenvolvido;
- Lista com os componentes que foram utilizados;
- Descrição da função de cada componente no circuito;
- Descrição das estratégias de implementação utilizadas e do programa elaborado.

2ª Parte: A segunda etapa da aula deve ser reservada para a apresentação dos projetos. Os estudantes devem fazer a apresentação de slides e demonstrar o projeto funcionando.

Os estudantes completaram mais uma etapa de sua incursão no universo da Computação. Por isso, é importante ofertar um clima de festividade, se possível convidando toda a comunidade a participar das apresentações. É possível reservar um outro momento para apresentação para a comunidade, reservando o restante da aula para o ensaio das apresentações.